Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Execução de Workflows Científicos na Plataforma BioNimbuZ

Pedro Gabriel Morais Angelo
Rômulo Pires Saraiva

Monografia apresentada como requisito parcial
para conclusão do Curso de Computação — Licenciatura

Orientadora
Prof.a Dr.a Aletéia Patrícia Favacho de Araújo

Brasília
2018

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Execução de Workflows Científicos na Plataforma BioNimbuZ

Pedro Gabriel Morais Angelo
Rômulo Pires Saraiva

Monografia apresentada como requisito parcial
para conclusão do Curso de Computação — Licenciatura

Prof.a Dr.a Aletéia Patrícia Favacho de Araújo (Orientadora)
CIC/UnB

Prof.a Dr.a Maristela Terto de Holanda
CIC/UnB

Prof. Edward Ribeiro de Oliveira
Senado Federal

Prof. Dr. Wilson Henrique Veneziano
Coordenador do Curso de Computação — Licenciatura

Brasília, 12 de dezembro de 2018

Dedicatória

Eu, Pedro Gabriel, dedico este trabalho aos amigos que compartilharam momentos e experiências durante toda a graduação, aos meu amigos de infância, aos meus sogros, Divino e Marlene, aos meus pais, Aquiles e Maria e minha namorada, Walessa, que acompanharam de perto esta experiência, me incentivaram e estiveram do meu lado nos bons momentos e nos momentos mais difíceis. Eu, Rômulo, dedico este trabalho à minha família, meus pais, Raimundo e Neide, meus irmãos Elen, Guilherme e Mariana, e à minha namorada, Isabella, que me deram força e coragem em todos os momentos. Aos meus amigos, pelas alegrias, tristezas e dores compartilhadas. A todos aqueles que de alguma forma estiveram e estão próximos de mim, fazendo esta vida valer cada vez mais a pena.

Agradecimentos

Agradecemos aos nossos pais, que ofereceram apoio e carinho nessa etapa decisiva da vida acadêmica. Somos gratos aos amigos de graduação, que tornaram os dias de aula mais felizes. Gratidão eterna à nossa orientadora Aletéia Patricia, responsável pela realização deste trabalho. Aos amigos Breno, Felipe, Jefferson e Fabiana pelo incentivo e grande ajuda para a realização deste trabalho.

Resumo

A computação em nuvem surge com a proposta de virtualizar e provisionar recursos computacionais, tais como processamento e armazenamento, com cobrança *pay-per-use* e utilização sob demanda, proporcionando uma economia de recursos. Além disso, tem-se a federação de nuvens, que soluciona o desafio de fornecer uma grande quantidade de recursos, na qual uma única nuvem não seria capaz de oferecer, possibilitando realizar tarefas complexas, como execução de *workflows* científicos. Para isso, tem-se a plataforma de nuvens federadas híbridas BioNimbuZ, que é um sistema voltado para a execução de *workflows* científicos, utilizando os benefícios fornecidos pela federação de nuvens, mas que atualmente foi usada apenas para executar *workflows* de Bioinformática. Assim, este trabalho objetiva demonstrar que a plataforma BioNimbuZ possui estrutura para executar *workflows* científicos de várias áreas de pesquisa. Para isso, foram selecionados três *workflows* científicos, os quais são das áreas de Astronomia, Bioquímica, e Reconhecimento Facial. Verificou-se que a plataforma é capaz de implementar e realizar tarefas referentes a diferentes *workflows* científicos, atentando-se apenas para a instalação de bibliotecas necessárias na instância antes da execução das mesmas. Além disso, não é necessário realizar grandes adaptações ou mudança na regra de utilização da plataforma. Assim, a plataforma BioNimbuZ provou ser flexível e eficiente, podendo ser usado para a execução de *workflows* de diferentes áreas.

Palavras-chave: Computação em nuvem, Nuvens federadas, *Workflows* científicos, BioNimbuZ.

Abstract

Cloud computing have the proposal to virtualize and provision computing resources, such as processing and storage, with pay-per-use and on-demand methods, providing resource savings. In addition, it has the cloud federations, which solves the problem of providing a large amount of resources, which a cloud is not able to offer, making it possible to perform complex tasks such as the execution of scientific workflows. To execute it, there is the BioNimbuZ, a hybrid federated clouds platform, which is a system for the execution of scientific workflows, using the cloud federation windows, but which currently have only a Bioinformatics workflow. Thus, this work purposes that BioNimbuZ platform has the structure to execute scientific workflows of several areas of research. To do this, three scientific workflows, which are from the areas of Astronomy, Biochemistry and Facial Recognition. It was verified that the platform is able to execute and perform tasks referring to various scientific workflows, paying attention only to a library installation that are executed before the execution of the task. In addition, there is no need to make adaptations. Thus, the BioNimbuZ platform has proved to be flexible and efficient and can be used to execute workflows from different areas.

Keywords: Cloud computing, Federated clouds, Scientific workflows, BioNimbuZ.

Sumário

Lista de Figuras

Capítulo 1

Introdução

Há pouco tempo tem surgido e desenvolvido na computação um modelo capaz de melhorar o poder de processamento e de armazenamento, voltado para a utilização da Internet, da virtualização e uso otimizado de recursos computacionais, proporcionando uma redução de gastos [14]. Este modelo é denominado como computação em nuvem [15], que possui características como virtualização, elasticidade, escalabilidade, que são necessárias para definir os serviços ofertados por provedores de nuvens.

Para utilizar os recursos da computação em nuvem, o usuário necessita de conexão com a Internet para ter acesso a aplicações ou estruturas, sendo fornecido poder computacional de acordo com a necessidade, ou seja, é oferecido processamento e armazenamento de forma virtualizada e escalável, provisionando mais poder computacional ou liberando-o, se necessário. O método de cobrança é sob demanda, no qual o usuário paga apenas pela quantidade de recursos que foi utilizado, que é comum em várias serviços oferecidos, como fornecimento de luz e de água [16]. No entanto, o modelo de computação em nuvem não fornece recursos infinitos, pois os provedores que fornecem o serviço possuem poder computacional limitado. Assim, casos de esgotamento de recursos podem ocorrer, principalmente, quando o usuário necessita utilizar grandes quantidades de processamento e de armazenamento.

Diante deste cenário, surgiu o modelo computacional de federeção de nuvem, que propõe a interoperabilidade de mais de uma nuvem, resolvendo o problema de limitação e estendendo seus recursos, além de apresentar uma redução de custos ao integrar nuvens. Assim, os provedores de nuvens podem fornecer seus recursos não utilizados por meio de uma interface que se conecta a outros provedores. Com isso, o usuário usufrui de um aumento do poder de processamento e da capacidade de armazenamento, de forma transparente[12].

Existem diversas formas de uma federação de nuvens operar. Por isso, na literatura, são apresentadas diferentes arquiteturas para a implementação de nuvens federadas [10,

12, 13, 17]. Entre elas, surgiu a plataforma de nuvem federada BioNimbuZ, um sistema voltado para execução de *workflows* científicos.

Um *workflow* científico trata-se de um conjunto de tarefas que possuem uma determinada dependência entre si, com uma finalidade científica a ser alcançada. O seu objetivo é automatizar essas tarefas, que geralmente são de análise ou de transformação de dados, pois tais tarefas, se realizadas separadamente e de forma manual, estarão mais propensas a erros, além de se tornarem tarefas dispendiosas. Existem *workflows* de váriadas áreas científicas como, por exemplo, Visão Computacional, Astronomia, Bioinformática e Bioquímica [18].

As tarefas realizadas por um *workflow* científico podem utilizar muito recurso computacional, pelo fato de trabalharem com grandes quantidades de dados. Assim, a utilização da plataforma BioNimbuZ é necessária, pois oferece suporte para a execução de *workflows* em nuvens federadas híbridas, de maneira simples para o usuário. Porém, atualmente, o BioNimbuZ foi usado para executar apenas *workflows* da área de Bioinformática.

Neste contexto, este trabalho propõe apresentar *workflows* científicos de diferentes áreas e executá-los no BioNimbuZ, realizando adaptações se necessário, porém, mantendo a proposta original da plataforma. Assim, será possível afirmar que o BioNimbuZ é uma plataforma de nuvens federadas híbridas para a execução de *workflows* científicos de qualquer área, não apenas *workflows* de Bioinformática.

1.1 Problema

O BioNimbuZ, desde a proposta de sua arquitetura por Saldanha [17], passando pelas suas melhorias [4, 19, 20, 21], até a arquitetura de sua versão mais atual proposta por Lopes e Gomes [10, 14], possuiu tarefas necessárias para execução de *workflows* científicos da área de Bioinformática.

Pelo fato de existir *workflows* científicos de diversas áreas, como Bioquímica, Astronomia e Visão Computacional, não é possível determinar se há desafios relacionados a implementação e execução de tarefas de *workflows* a serem superados, nem mapeá-los caso haja. Assim, não há garantias de que a plataforma BioNimbuZ pode realizar a execução de qualquer *workflow*. E como os *workflows* científicos, independentes da área, podem se beneficiar da execução em plataformas de nuvens, este projeto propõe mostrar que a plataforma BioNimbuZ é capaz de atender diferentes demandas das áreas científicas.

1.2 Objetivo

O objetivo principal deste trabalho é demonstrar que a plataforma de nuvem federada híbrida BioNimbuZ permite a implementação de tarefas, realizando a execução de *workflows* científicos de qualquer área de pesquisa.

Para atingir o objetivo principal, é necessário atingir os seguintes objetivos específicos:

- Definir três *workflows* científicos de áreas de pesquisa diferentes para serem implementados na plataforma BioNimbuZ;
- Analisar os *workflows* definidos, identificando bibliotecas e requisitos necessários para sua execução;
- Realizar adaptações na plataforma, visando a implementação dos *workflows* a serem estudados;
- Executar cada *workflows* científico na plataforma BioNimbuZ, acompanhando a execução de cada tarefa;
- Avaliar os resultados obtidos da execução dos *workflows* científicos, analisando seus respectivos arquivos de saída.

1.3 Organização do Documento

Este documento possui, além deste capítulo de introdução, mais cinco capítulos. No Capítulo 2 são definidos os conceitos de *workflows* científicos, além de apresentar os três *workflows* a serem utilizados neste trabalho.

No Capítulo 3 são apresentadas a computação em nuvem, as suas características principais e modelos. Além disso, é definido o conceito de nuvem federada, suas características e algumas arquiteturas existentes.

No Capítulo 4 é apresentada a plataforma de nuvem federada BioNimbuZ, sua proposta e arquitetura, além de mostrar os fluxos de execução de um *workflow* na plataforma.

O Capítulo 5 apresenta os resultados obtidos da execução dos três *workflows* científicos no BioNimbuZ. Por fim, o Capítulo 6 apresenta a conclusão deste trabalho e os futuros trabalhos.

Capítulo 2

Workflows Científicos

Este capítulo apresenta uma visão geral a respeito de *workflows* científicos, assim como detalha quatro *workflows* de diferentes áreas científicas. Para isso, a Seção 2.1 apresenta as definições e as características de um *workflow* científico. A Seção 2.2 aborda o *workflow* científico que trabalha com dados de Bioinformática, já presente na plataforma BioNimbuZ. As Seções 2.3, 2.4 e 2.5 apresentarão os *workflows* das áreas de Astronomia, Bioquímica e Reconhecimento Facial, respectivamente, que serão utilizados neste trabalho.

2.1 Conceitos de *Workflows* Científicos

Um *workflow* científico é um conjunto de tarefas ordenadas que atuam para uma mesma finalidade. Cada tarefa representa uma etapa do *workflow*, que podem ser relacionadas com outras etapas, de acordo com suas dependências, ou seja, quando o dado de saída de uma determinada etapa é necessária para a etapa seguinte [18].

De acordo com Braghetto [22], um *workflow* científico pode ser definido como "uma automação de um experimento ou de um processo científico, expressa em termos das atividades a serem executadas e, principalmente, das dependências dos dados manipulados". Braghetto [22] também afirma que *workflows* são formados por dados de entrada e de saída, análises e ferramentas ordenadas para que o objetivo seja alcançado. Já Singh *et al.* [23] afirmam que o termo *workflow* científico descreve uma série de atividades e cálculos estruturados, cujo objetivo é resolver problemas científicos.

O objetivo de um *workflow* científico é automatizar tarefas que normalmente se repetem, que acessam dados, realizam transformação ou análise e que, se feitas manualmente ou de maneira independente, estariam mais propensas a erros [3].

Existem *Workflows* científicos de diversas áreas. Neste trabalho, foram escolhidos o *workflow* Montage, de Astronomia; um *workflow* de Bioquímica; e um *workflow* de Reconhecimento Facial. O *workflow* Montage [24] é da área de Astronomia e utilizado

pela NASA. O *workflow* da área de Bioquímica foi desenvolvido e utilizado pela UNIFEI (Universidade Federal de Itajubá - Minas Gerais). Por fim, temos o *workflow* de reconhecimento facial, utilizado pelo Departamento CIC-UnB (Ciência da Computação da Universidade de Brasília).

Além dos *workflows* a serem utilzados neste trabalho, tem-se, por exemplo, *workflows* de outras áreas, como o *CyberShake* [1, 25], representado pela Figura 2.1. Ele é utilizado pelo Centro de Terremotos do Sul da Califónia (*Southern California Earthquake Center*), para calcular os riscos de terremotos por meio da Análise de Perigos Sísmicos Probabilísticos (*Probabilistic Seismic Hazard Analysis* - PSHA) [26].

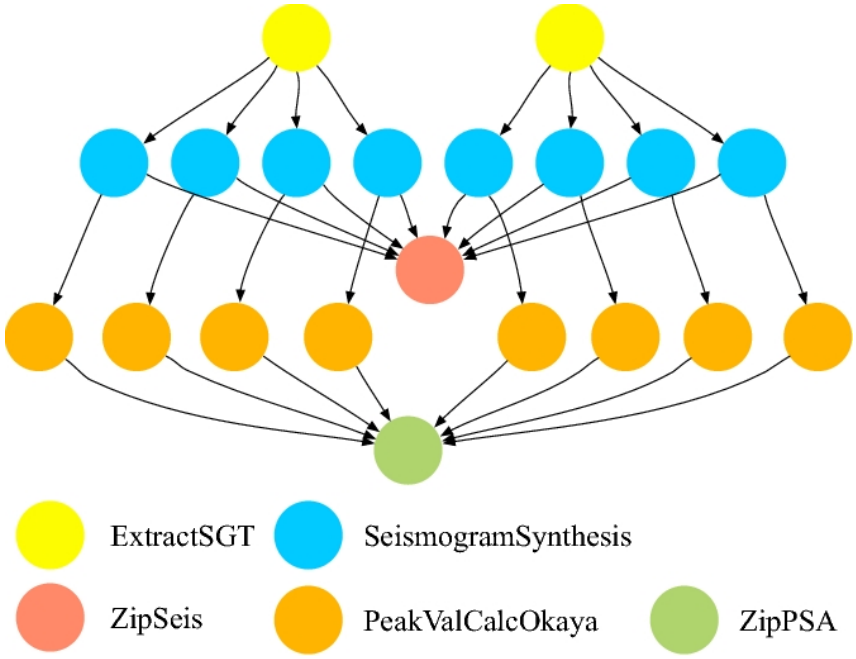

Figura 2.1: *Workflow Cybershake* [1].

A Figura 2.2 mostra um outro *workflow*, o *workflow Epigenomics*, que é utilizado pelo *University of Southern California* Epigenome Center [27] para realizar o mapeamento de células humanas em escalas genômica. Assim, é possível notar a diversidade de áreas do conhecimento que fazem uso do conceito de *workflows*.

2.1.1 Estruturas Básicas

Em um *workflow* científico, de acordo com o relacionamento das tarefas com os dados de entrada e de saída, é possível criar várias combinações entre eles. Além disso, os dados de saída de uma tarefa pode se tornar dados de entrada de outra tarefa, de acordo com sua dependência.

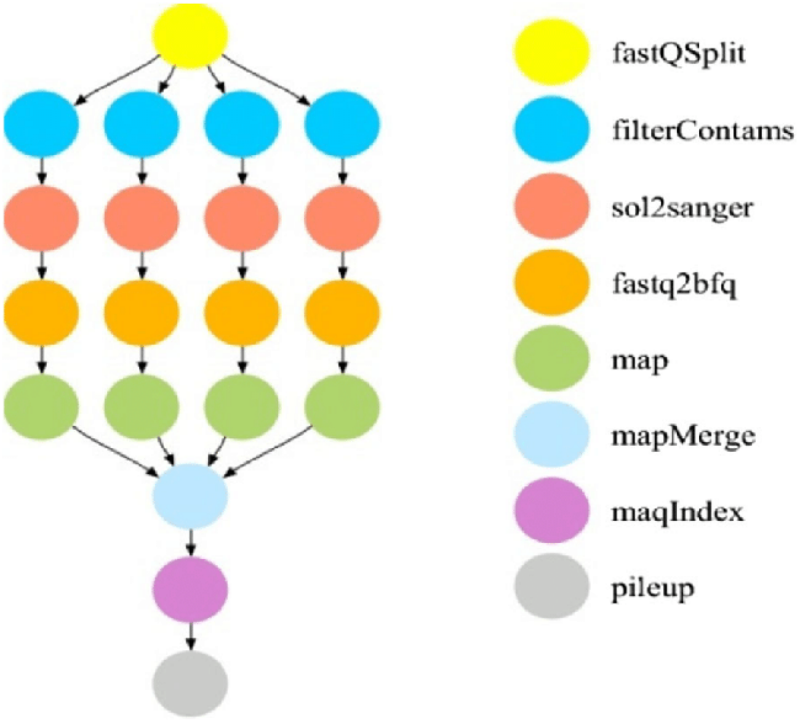

Figura 2.2: *Workflow Epigenomics* [1].

Por isso, segundo Bharathi [2], existem algumas estruturas básicas em um *workflow*, conforme é apresentado na Figura 2.3.

A primeira estrutura é denominada de Processo. Trata-se apenas da realização de uma tarefa com uma entrada que gera uma saída. A segunda estrutura chama-se *Pipeline*, e trata-se de uma sequência de Processos, no qual a saída gerada pela primeira tarefa será a entrada da segunda tarefa que, por sua vez, irá gerar uma única saída. O terceiro modelo, chamado de Distribuição de Dados, configura a realização de uma tarefa com uma entrada e fornecendo várias saídas. Por outro lado, a estrutura chamada de Agregação de Dados fornece uma única saída, porém, necessita de várias entradas para realizar a tarefa.

Por fim, a estrutura chamada Redistribuição de Dados trata-se de uma combinação das estruturas Agregação de Dados e Distribuição de Dados, ou seja, para que a tarefa possa ser realizada é necessário que sejam fornecidas várias entradas e, após a conclusão da tarefa, várias saídas serão geradas. Esta estrutura, assim como as estruturas Agregação de Dados e Distribuição de Dados, representa o paralelismo no *workflow*, pois o fornecimento de arquivos de saída para várias tarefas a serem executadas posteriormente indicam que elas são independentes, ou seja, elas podem ser executadas simultaneamente. Por outro lado, quando uma tarefa recebe vários arquivos de entrada, significa que as tarefas realizadas anteriormente que geraram estas entradas foram executadas de forma paralela.

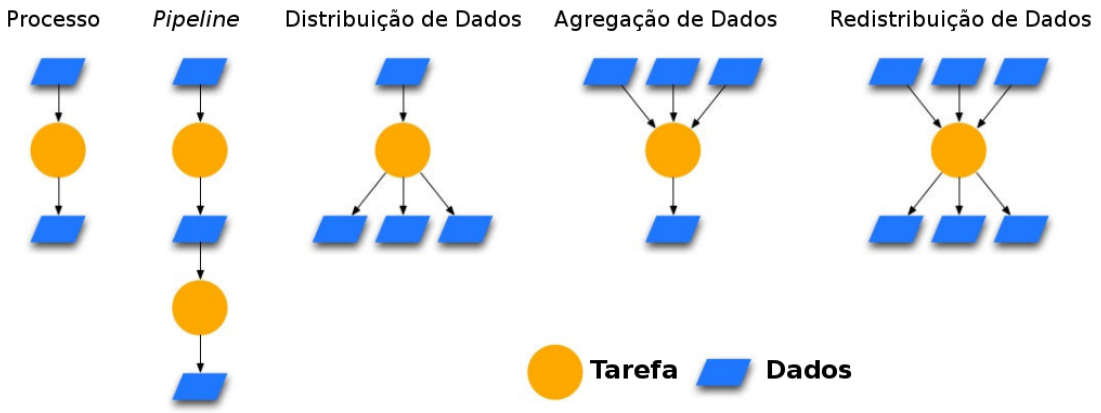

Figura 2.3: Representações de Estruturas Básicas de um *Workflow* [2].

2.1.2 Ciclo de Vida de um *Workflow*

Para executar um *workflow* científico é necessário passar por algumas etapas em relação ao desenvolvimento, implantação e execução, definindo assim um ciclo de vida. Assim, um ciclo é formado em cinco fases, conforme apresentado na Figura 2.4, que são [22]:

- **Projeto e Composição:** nesta fase o *workflow* começa a ser desenvolvido, geralmente a partir do levantamento de requisitos, baseado nas hipóteses da pesquisa a serem testadas ou finalidade a ser alcançada. Em seguida, são criadas especificações das tarefas que irão compor o *workflow* e, assim será criado o próprio *workflow*. Dessa forma, o pesquisador pode utilizar um *workflow* já existente ou reutilizar algumas tarefas a partir do repósitorio de *workflows*;
- **Planejamento de Recurso:** esta é a fase de preparação de recursos para a execução do *workflow*. Nesta fase ocorre a programação, a otimização e a seleção dos dados de entrada, além da definição do sistema a ser utilizado e a alocação de outros recursos;
- **Execução:** é a fase de realização das etapas do *workflow* científico nos recursos alocados, com a inserção dos dados de entrada no *workflow*, e o seu processamento em uma plataforma de computação. No entanto, durante a execução, o cientista tem a possibilidade de acompanhar os dados intermediários, para que possa corrigir, posteriormente, possíveis problemas na execução. Tanto os dados de entrada quanto os dados intermediários são armazenados no repositório de dados;
- **Análise da Execução:** esta fase trata da inspeção e da interpretação dos dados de saída após a execução do *workflow*, por meio da ferramenta que realizou o monitoramento durante a execução. Após a análise, é possível validar a hipótese ou não, verificar o passo a passo da execução e analisar o desempenho do *workflow*;

- **Compartilhamento dos Resultados:** as análises e os dados dos resultados podem ser publicados entre o grupo de pesquisadores envolvidos e compartilhados. Assim, se necessário for, os pesquisadores podem fazer as adequações na hipótese ou no objetivo do *workflow*, e realizar um reprojeto, dando início a um novo ciclo.

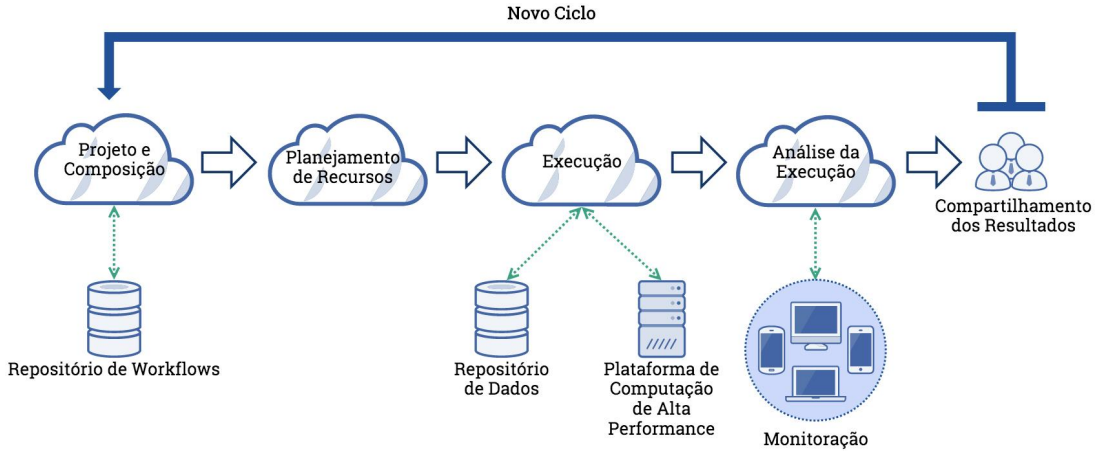

Figura 2.4: Detalhamento do Ciclo de Vida de um *Wokflow* Científico [3].

Diante do exposto, é possível notar que os *workflows* são usados em diferentes áreas. Neste trabalho será exemplificado o uso de *workflows* em quatro áreas do conhecimento, os quais serão descritos em detalhes nas próximas seções.

2.2 *Workflow* de Bioinformática

O *workflow* de Bioinformática trabalha com dados biológicos, tendo como objetivo o sequenciamento de DNA (Ácido Desoxirribonucleico) ou RNA (Ácido Ribonucleico), para descobrir a base de cada fragmento. Esse processamento possui duas categorias, que são: projeto genoma para estudos do DNA e projeto transcritoma para estudos do RNA [4].

Para realizar esse sequenciamento, tanto o projeto genoma quanto o projeto transcritoma possuem condicões computacionais para que seja possível fragmentar os dados de entrada e obter informações biológicas. Para chegar nesse objetivo é apresentada na Figura 2.5 um modelo de *workflow* de Bioinformática com três tarefas [4]: filtragem, mapeamento/montagem e análise. Essas etapas serão descritas em detalhes a seguir:

- **Filtragem:** esta fase ocorre após o pesquisador realizar a coleta e a replicação de material biológico e, após isso, sequenciá-lo em pequenas porções de DNA ou RNA de variados tamanhos. Essas pequenas porções são denominadas *reads*, que são formadas por bases nitrogenadas, composto por Adenina, Citosina, Guanina e Timina. Como as *reads* possuem qualidades diferentes, durante a etapa de filtragem

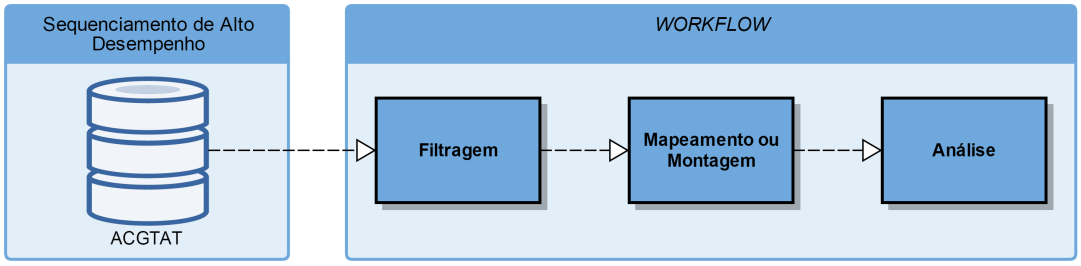

Figura 2.5: Exemplo de *Workflow* para Projetos Genoma e Transcritoma [4].

é realizada a seleção de *reads*, eliminando aqueles de qualidade inferior. Além disso, nessa etapa também são identificados erros laboratoriais. Como é uma etapa de complexidade baixa, ela pode ser realizada por *scripts* desenvolvidos pelo próprio usuário, mas também há ferramentas como FASTX-*toolkit* [28] para executar essa tarefa;

- **Mapeamento/Montagem:** quando o pesquisador conhece o genoma de referência, é realizada a etapa de mapeamento, no qual as *reads* filtradas na etapa anterior são encontradas e alinhadas, agrupando-as em conjuntos maiores, apresentado na Figura 2.6. O mapeamento, normalmente, é utilizado em organismos que já possuem seu DNA sequenciado. Uma ferramenta utilizada para o mapeamento é o Bowtie [29], que é considerada rápida, pelo fato de conseguir alinhar mais de 25 milhões de *reads* com o consumo de memória de 1,3Gb.

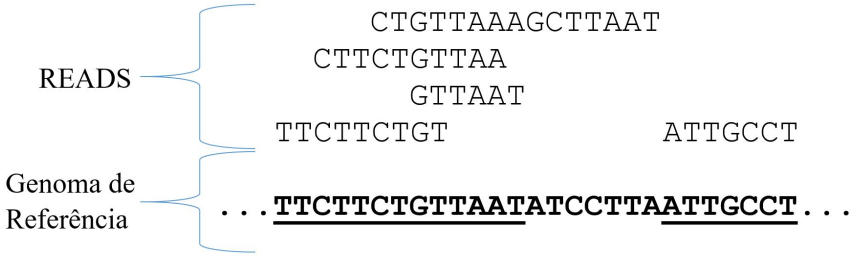

Figura 2.6: Exemplo de Mapeamento [5].

Quando o genoma de referência é desconhecido, é executada a etapa de montagem, na qual é possível realizar um alinhamento de *reads*, gerando conjuntos maiores denominados *contigs*. Para descobrir o conjunto do cromossomo sequenciado é feito um alinhamento de *reads*, e o resultado é apresentado por meio de sua sombra, de acordo com a Figura 2.7 [5];

- **Análise:** nesta fase o pesquisador possui uma grande quantidde do DNA ou do RNA mapeado, possibilitando sua análise de acordo com o seu objetivo. Um exemplo de finalidade para esta etapa, citado por de Paula [5], é a verificação de suscetibilidade

Figura 2.7: Exemplo de Montagem [5].

a um determinado vírus, por meio de diferenças apresentadas na expressão genética. Para isso, é realizado o mapeamento de amostras de RNA de seres suscetíveis e não suscetíveis, realizando uma comparação dos transcritomas na fase de análise.

2.3 *Workflow* de Astronomia

O *workflow* científico da área de astronomia a ser descrito nesta seção é o Montage [30]. Ele foi desenvolvido pela NASA/IPAC (*Infrared Processing and Analysis Center*), ligado à Caltech (*California Institute of Technology*), utilizado para criar mosaicos a partir de imagens produzidas do céu [24].

A sua primeira versão foi lançada em 2003 e, nas três primeiras versões, apresentadas entre 2003 e 2010, o *workflow* montava mosaicos apenas em duas dimensões. A partir da quarta versão, lançada em 2015, tornou-se possível criar mosaicos multidimensionais. Na quinta versão, o Montage passou a processar dados do tipo HEALPix (*Hierarquical Equal Area Longitude Pixelization*) e TOAST (*Tessellated Octahedral Adaptive Subdivision Transform*), possibilitando a integração com o *World Wide Telescope*, um *software* que simula um telescópio [31] [32], além de possibilitar a utilização do *workflow* como biblioteca. Ainda está previsto para as duas próximas versões, cinco e seis, a execução do *workflow* no sistema operacional Windows [33], e como biblioteca para a linguagem de programação Python [34], respectivamente [6].

O Montage executa uma série de processos para que o conjunto de imagens se torne um mosaico. No exemplo apresentado na Figura 2.8 é feita uma análise do tamanho das imagens FITS (*Flexible Image Transport System* [32]), para que seja possível obter as dimensões do mosaico final. Em seguida, é realizada a reprojeção das imagens de entrada, seguido de uma retificação de fundo, um alinhamento, retirando as variações entre as imagens e, após isso, é realizada a modelagem do fundo do mosaico. Por fim, todas as imagens são agrupadas uma ao lado da outra, formando o mosaico. Esse último processo é chamado de co-adição [6].

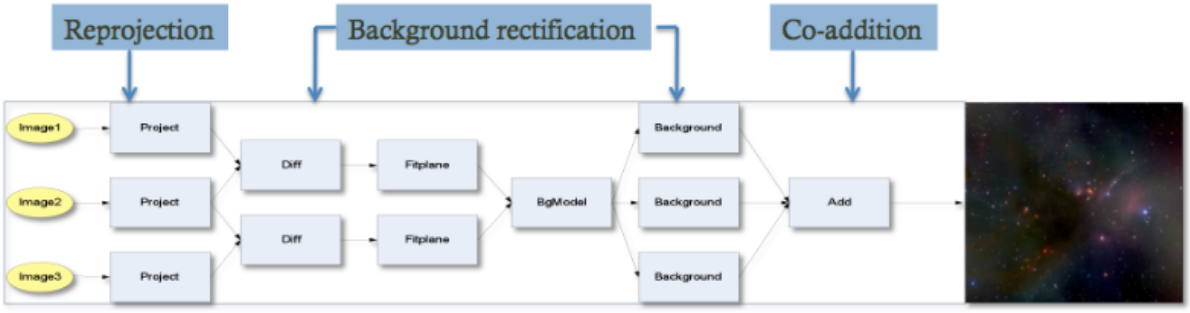

Figura 2.8: Modelo do *Workflow* Científico Montage. [6].

O *workflow* Montage [35] a ser executado neste trabalho serão *scripts* na linguagem *Shell Script* [36]. No *workflow* são realizados os três processos de acordo com o modelo apresentado na Figura 2.8 (reprojeção, retificação de fundo e co-adição), que podem ser executados por uma ou mais tarefas. Assim, o *workflow* Montage que será implementado no BioNimbuZ possui dez tarefas distintas, que são [35]:

- **mImgtbl:** esta tarefa realiza a extração de informações de geometria do cabeçalho de um conjunto de arquivos FITS, e criada uma tabela de metadados para ser utilizado por outras etapas. Assim, além da extração e da criação da tabela, esta tarefa aponta possíveis falhas de verificação;
- **mMakeHdr:** esta tarefa gera um cabeçalho dos arquivos FITS que irão descrever a imagem de saída. O cabeçalho é gerado através dos metadados de entrada (a tabela da tarefa mImgtbl) e baseado em suas informações. Se os metadados de entrada refletirem um conjunto de imagens combinados de forma adequada, o cabeçalho irá representar no mosaico;
- **mProjExec:** esta tarefa realiza uma reprojeção para cada imagem que está na tabela de metadados, de acordo com a escala definida no arquivo de cabeçalho. Em seguida, é criado um arquivo de metadados das imagens reprojetadas, estas que estão armazenadas em uma pasta;
- **mAdd:** tarefa na qual é realizada uma co-adição das imagens reprojetadas para que um mosaico seja formado na saída. Para isso, é necessário indicar a pasta onde está localizada as imagens reprojetadas e o cabeçalho dos arquivos FITS. Na saída, são gerados dois arquivos, um representando o arquivo FITS, e outro arquivo contendo valores de área de *pixel* co-adicionados;

- **mViewer:** é a etapa na qual é gerado um arquivo de imagem JPEG (*Joint Photographics Experts Group*) a partir de um arquivo de entrada no formato FITS, mostrado na Figura 2.9, ou três arquivos FITS em cores. Para a imagem é possível alterá-la por meio de argumentos fornecidos para a função *log* (na qual o zero é para imagem linear), ou usando algoritmos de equalização de histograma gaussianos [37];

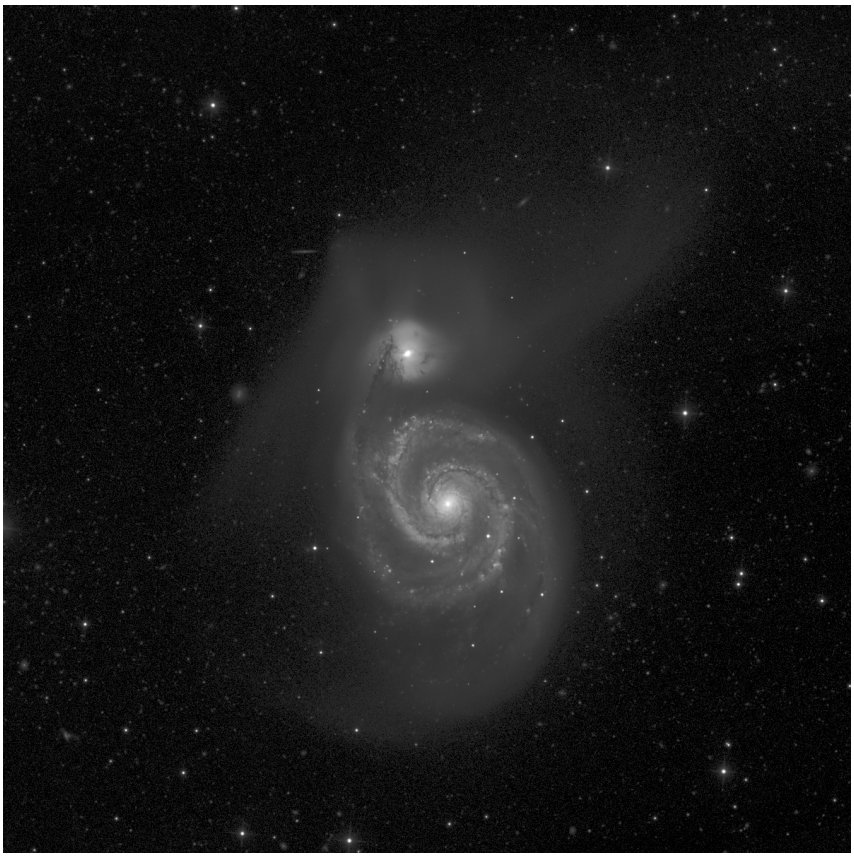

Figura 2.9: Exemplo de Mosaico Produzido pela Tarefa mViewer [7].

- **mOverlaps:** nesta etapa é realizada uma análise da tabela de metadados (arquivo de saída da tarefa mImgtbl), para gerar uma nova tabela com informações de imagens sobrepostas. Cada imagem é comparada com todas as outras para determinar os pares de imagens sobrepostas;

- **mDiffExec:** esta tarefa realiza um cálculo de diferença simples entre cada par de imagens sobrepostas, na qual foi identificado pela tabela com informações de imagens sobrepostas (arquivo de saída da etapa mOverlaps);

- **mFitExec:** esta etapa ajusta mínimos quadros em um plano dentro de uma imagem. Esse ajuste é realizado em todas as diferenças que foram identificadas pela etapa mOverlaps, e geradas pela etapa mDiffExec (no qual está localizado em uma pasta, passado por argumento);

- **mBgModel:** é a tarefa na qual, ao utilizar a tabela de parâmetros de diferença de imagem pra imagem (arquivo de saída gerada pela etapa mFitexec), determina um conjunto de correções a serem aplicadas a cada imagem para obter o melhor ajuste. Assim, é realizada uma modelagem de plano de fundo;
- **mBgExec:** esta tarefa aplica as correções em todas as imagens que foram indicadas pela tarefa mBgModel, ou seja, pelo seu arquivo de saída com a relação de metadados. A correção de fundo é aplicada à imagem especificada como Ax + By + C, onde as coordenadas (x,y) representam os *pixels*, sendo o centro da imagem como origem, enquanto (A,B,C) corresponde aos parâmetros do plano de fundo.

2.4 *Workflow* de Bioquímica

O *workflow* de Bioquímica foi desenvolvido pelo Instituto de Ciências Tecnológicas da Universidade Federal de Itajubá, no Campus Itabira (ICT-UNIFEI), em parceria com o Departamento de Computação da Universidade de Brasília (CIC-UnB). Este *workflow* visa preparar uma dinâmica molecular, que é uma simulação computacional que permite estudar o movimento de moléculas, possibilitando que seja observado seu comportamento no ambiente em que as moléculas estarão inseridas [38].

O *workflow* é formado por *scripts* na linguagem de programação Python [34] e possui três etapas, conforme apresentado na Figura 2.10. A primeira etapa realiza uma preparação, retirando informações como cabeçalho e outros campos, ou seja, formatando o arquivo de entrada no formato PDB (*Protein Data Bank*), que é o arquivo que representa uma proteína. Esta tarefa tem como objetivo reduzir o tempo de processamento e gastos com memória nas tarefas posteriores, pois elas percorreriam partes do arquivo PDB que são desnecessárias para a pesquisa. Neste trabalho, foi utilizada a proteína 1t80 [8], apresentada na Figura 2.11, encontrada na plataforma RCSB[1]. Após a execução desta tarefa, como arquivo de saída, é fornecido um arquivo no formato PDB, formatado, somente com as informações a serem analisadas pelas tarefas seguintes.

A segunda etapa realiza a retirada das moléculas de água cristalográfica na estrutura molecular, para que não tenha interferência destas moléculas na análise do movimento molecular. Esta etapa pode ser optativa para o estudo do movimento da molécula, dependendo da decisão do pesquisador se deve ou não considerar as águas cristalográficas presentes na estrutura. Esta tarefa necessita de somente um arquivo de entrada no formato PDB e fornece um arquivo de saída no mesmo formato, porém, sem as águas cristalográficas.

[1]https://www.rcsb.org/structure/1t80

Por fim, a terceira etapa consiste em verificar *gaps* das moléculas. Esta tarefa percorre todo o arquivo PDB de entrada, verificando as ligações dos átomos, localizando trechos com numerações repetidas, realizando uma nova numeração ou selecionando um dos trechos para permanecer no arquivo. Após a execução da tarefa, é fornecido como saída um arquivo no formato PDB, com as correções dos *gaps* encontrados.

Figura 2.10: Modelo do *Workflow* de Bioquímica.

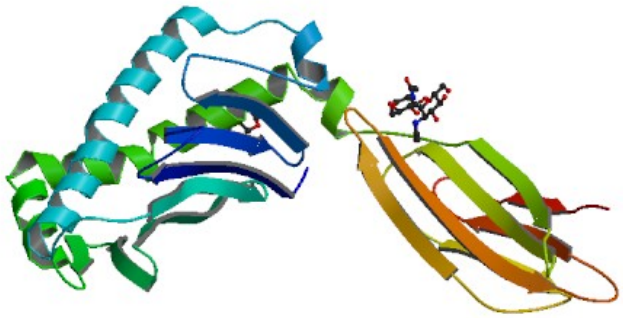

Figura 2.11: Molécula 1t80 [8].

2.5 *Workflow* de Reconhecimento Facial

O *workflow* a ser descrito nesta seção é o *workflow* de Reconhecimento Facial, utilizado pelo Departamento de Ciência da Computação da Universidade de Brasília [39]. Este

workflow foi desenvolvido por meio da biblioteca OpenCV [9] (*Open Source Computer Vision* ou visão computacional de código aberto), que permite manipular imagens e vídeos para realizar várias tarefas, como reconhecer faces de uma *webcam*, como apresentado na Figura 2.12, até ensinar um robô a reconhecer objetos [9].

O *workflow* utilizado neste trabalho possui duas etapas, conforme apresentado a Figura 2.13. A primeira etapa consiste em receber uma imagem como arquivo de entrada, identificar possíveis faces e armazená-las em uma pasta, que será fornecida como arquivo de saída [39]. Nesta pasta, todas as faces reconhecidas pela tarefa são armazenadas com o padrão de escala de cinza, e com o tamanho de 200x200 *pixels*.

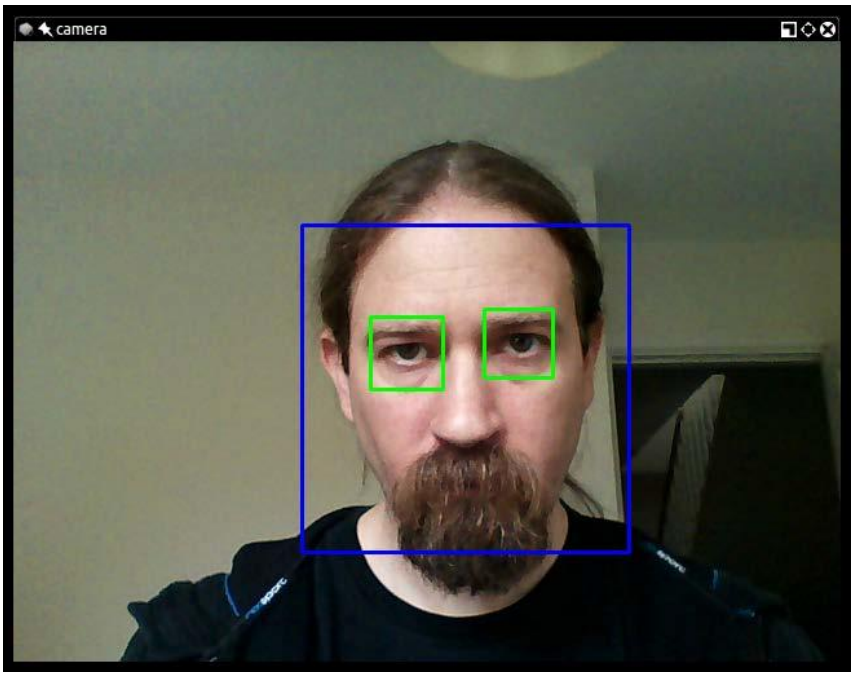

Figura 2.12: Reconhecimento Facial Realizado pelo OpenCV [9].

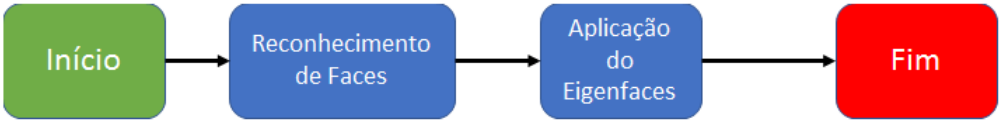

Figura 2.13: Modelo do *Workflow* de Reconhecimento Facial.

A segunda etapa recebe uma imagem e a pasta com amostras de faces originado pela etapa anterior e realiza o reconhecimento facial, comparando a imagem de entrada com a pasta de amostras. Esta tarefa reconhece as faces do arquivo de entrada, transforma-as em imagens de escala de cinza e tamanho 200x200 *pixels*, assim como as imagens da

pasta de amostras que foram originadas pela pasta de saída da primeira tarefa, para que seja realizada a comparação. A saída desta etapa fornece a imagem de entrada com a face identificada e um arquivo de *log*, informando a amostra comparada e a pontuação adquirida para a comparação [39].

O algoritmo de comparação utilizado chama-se Eigenfaces, um método baseado na busca de auto-vetores da matriz de covariância de um conjunto de imagens de faces [40]. O seu funcionamento consiste em realizar a análise dos principais componentes da imagem, comparando com as amostras de faces e, após a análise, é determinado se a face foi identificada e, além de ser atribuído um *score* de confiança, no caso de identificação, no qual quanto menor o valor, menor será a diferença entre as imagens analisadas. Caso seja atruibuído o *score* zero, significa que a imagem é idêntica a amostra analisada [9].

Além do Eigenfaces, existem mais dois algoritmos capazes de realizar comparações. O primeiro é o Fisherfaces, que é semelhante ao Eigenfaces, ou seja, após é realizado a comparação de imagens e atribuído um valor. No entanto, o Fisherfaces tende a produzir um valor mais preciso. O outro algoritmo é o LBPH (*Local Binary Pattern Histograms* ou Histogramas de Padrão Binário Local) que divide a face detectada em pequenas áreas e realiza uma comparação com a área correspondente à amostra, produzindo um valor para cada área comparada. Assim, este algoritmo permite que as faces das amostras e as faces detectadas sejam de diferentes formas e tamanhos [9].

2.6 Considerações Finais

Neste capítulo foram descritos conceitos de *workflows* científicos, as estruturas que compõem a organização de um *workflow* e o seu ciclo, ou seja, as etapas necessárias para que um *workflow* seja planejado, montado, executado e que seus resultados sejam analisados posteriormente. Além disso, foram apresentados exemplos de *workflows* científicos de diversas áreas, destacando os *workflows* que serão utilizados neste trabalho, como o *workflow* Montage, da área de Astronomia; o *workflow* de Bioquímica; e o *workflow* de Reconhecimento Facial.

Como apresentado neste capítulo, *workflows* científicos são um conjunto de atividades ordenadas, cujo objetivo é resolver um problema específico. Por isso, *workflows* científicos, de um modo geral, demandam uma alta capacidade de processamento. Assim, eles se beneficiam do uso de plataformas de nuvem computacional, que serão descritas no próximo capítulo.

Capítulo 3

Computação em Nuvem

Este capítulo objetiva apresentar os conceitos de computação em nuvem, por meio de definições, características e arquitetura. Além disso, serão descritos os modelos de implantação e os tipos de nuvens. Por último, são abordados a visão geral de federação de nuvens e dois modelos de arquitetura.

3.1 Visão Geral

A computação em nuvem proporcionou uma grande evolução na utilizção e na comercialização de hardwares e softwares no mercado. A sua popularidade se deve a possibilidade de provedores, através de seus *Data Centers*, fornecerem ao usuário o provisionamento de recursos computacionais, como processamento, memória e armazenamento de acordo com a demanda do cliente, ou seja, de forma escalável e ágil, dando-lhe a sensação de recursos infinitos [14]. O modelo de negócio utilizado é o *pay-per-use* [16], permitindo que o usuário pague de acordo com o seu uso, e isso geralmente garante uma melhor relação custo x benefício se comparado com custos de aquisição e manutenção de hardwares com características semelhantes, visto que tais recursos são obtidos virtualmente.

O Instituto Nacional de Padrões e Tecnologia, ou NIST (*National Institute of Standards and Technology*) [15], define computação em nuvem como um modelo que permite acesso de forma conveniente à rede de recursos de computação configuráveis, os quais podem ser redes, servidores, dispositivos de armazenamento, aplicativos ou serviços, rapidamente provisionados e liberados com pouco esforço de gerenciamento ou mínima interação com o provedor de serviço.

Já Ruschel *et al.* [41] afirmam que a computação em nuvem é a utilização de vários tipos de aplicações através da Internet, em qualquer lugar e independente da plataforma, com a facilidade semelhante a de possuí-las em nossos computadores, possibilitando que super-computadores sejam destinados a quem precisar, baseado na Internet.

Foster *et al.* [42], por sua vez, definem computação em nuvem como um paradigma computacional distribuído, direcionado por economia de escala, onde um *pool* de recursos computacionais disponibiliza de acordo com a demanda do usuário processamento, armazenamento, plataforma e serviços abstraídos, virtualizados dinamicamente e escaláveis para clientes por meio da Internet.

Portanto, é consenso entre as definições citadas que a computação em nuvem é um modelo, uma prática, na qual o usuário tem a possibilidade de utilizar grandes recursos computacionais, como processamento, armazenamento e serviços, virtualmente, através da Internet, com provisionamento e liberação de recursos de acordo com a demanda. Além disso, o cliente pode acessar tais recursos com facilidade, por meio de qualquer dispositivo, com pouca interação com o provedor de serviço, e a cobrança deve ser de acordo com os recursos e o tempo de utilização.

3.2 Características

Mell *et al.* [15], lista cinco características classificadas como essenciais para a computação em nuvem, que são:

- ***Self-service:*** o cliente pode provisionar os recursos computacionais, de forma unilateral, automática e conforme o necessário, sem a necessidade de intervenção humana com o provedor de serviços;
- **Amplo Acesso à Rede:** os recursos estão disponíveis para o cliente por meio da Internet, ou seja, promove o uso de dispositivos clientes com recursos próprios variados, de diversas plataformas, como *smartphones*, *tablets*, *laptops* e estações de serviço, por exemplo;
- ***Pooling* de Recursos:** os recursos oferecidos pelos provedores se agrupam para atender a múltiplos consumidores com variados dispositivos, de acordo com cada demanda, sem a necessidade de que cada cliente conheça a infraestrutura utilizada, e nem a sua localização geográfica;
- **Elasticidade Rápida:** os recursos podem ser provisionados ou liberados rapidamente de forma automática e sob a demanda do comsumidor, dando-o a sensação de recursos ilimitados;
- **Serviços Mensurados:** o sistema do plataforma de nuvem controla e otimiza os recursos automaticamente por meio da medição do tipo de serviço utilizado, promovendo transparência.

3.3 Modelos de Serviço

De acordo com NIST [15] e Pedrosa *et al.* [43], os modelos de serviço de computação em nuvem podem ser categorizados em três tipos:

- **SaaS - *Software as a Service*:** modelo que disponibiliza aplicações como serviço, que pode ser acessado por diversos usuários por meio da rede, sem que tenham conhecimento da infraestrutura envolvida para disponibilizá-lo;
- **PaaS - *Plataform as a Service*:** modelo que oferece uma plataforma que permite, além da disponibilização, o desenvolvimento de aplicações;
- **IaaS - *Infrastructure as a Service*:** modelo que fornece a infraestrutura física, como processamento, armazenamento, rede e outros recursos por meio da virtualização, além de permitir a personalização de diversas aplicações e sistemas operacionais.

Os três modelos possuem uma relação, conforme é apresentado na Figura 3.1. Os modelos podem ser organizados em camadas, representando suas dependências devido ao serviço que cada um fornece. Os serviços oferecidos pelo modelo SaaS pode ser implantados nos modelos Paas, IaaS ou nos recursos de hardware diretamente. Os serviços de PaaS podem ser implantados no modelo IaaS ou diretamente nos recursos de hardware, enquanto os serviços de IaaS, disponibilizados por meio de virtualização, são implantados nos recursos computacionais [10].

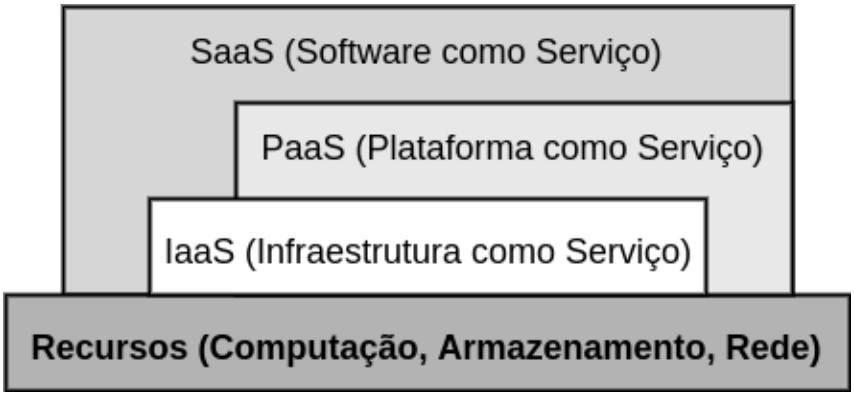

Figura 3.1: Arquitetura em Camadas da Nuvem, adaptado de [10].

3.4 Modelos de Implantação

Existem quatro modelos de implantação de nuvens computacionais, que varia de acordo com a necessidade da aplicação a ser oferecida, e o tipo de contrato de prestação de

serviço. Assim, os modelos são categorizados em Público, Privado, Comunitário e Híbrido, descritos a seguir [43]:

- **Nuvem Pública:** neste modelo a nuvem é disponibilizada para o público ou para grandes grupos industriais provisionar recursos. Ela é implementada por um prestador de serviço que deve ser capaz de garantir seu desempenho e sua segurança;
- **Nuvem Privada:** a nuvem é disponibilizada para uma determinada organização, no qual somente seus membros possuem acesso aos recursos. O gerenciamento da nuvem pode ser feito pela própria organização, por terceiros, ou de forma mista;
- **Nuvem Comunitária:** nesta modalidade a infraestrutura da nuvem é compartilhada por várias organizações, desde que tenham missão, requisitos de segurança e política em comum. Ela pode ser gerenciado pelas organizações ou por terceiros;
- **Nuvem Híbrida:** neste modelo, a infraestrutura é formada pela combinação de duas ou mais modalidades citadas anteriormente, na qual seja possível a portabilidade de dados ou aplicações. Por isso, cada modalidade permanece dentro de suas características.

3.5 Federação de Nuvens

A federação de nuvens é um modelo computacional o qual permite que vários provedores de nuvem possam integrar-se com outras nuvens, ou seja, fornece o compartilhamento de infraestrutura, por meio de acordos estabelecidos [12, 44]. O objetivo da federação de nuvens é buscar e alocar recursos, aumentando o poder computacional sem a interação de mediadores [45].

O modelo de federação em nuvens é indicado, por exemplo, para executar aplicações que precisam de grandes capacidades de recursos, ou seja, um provedor pode não ser suficiente para sua execução de forma eficiente. O provedor pode não ser capaz de provisionar recursos, ou não permitir que tal recurso seja fornecido a um determinado usuário, devido às políticas que impõem limitações ao usuário [46].

Toosi *et al.* [11] afirmam que após a formação da federação de nuvens, ela pode operar de duas formas, as quais são vista na Figura 3.2, e descritas a seguir:

- ***Multi-cloud:*** as nuvens fornecem recursos, mas sem saber se há outras nuvens fornecendo recursos também e vice-versa, ou seja, não há participação voluntária ao conectar as nuvens. Por isso, é necessário que seja feito o gerenciamento de recursos entre as nuvens;

- ***Intercloud:*** esta relação é definida como qualquer interação entre nuvens, incluindo *Multi-cloud*. Várias nuvens são conectadas entre si, formando uma rede. Assim, esta relação remove dificuldades relacionadas a migração de escala.

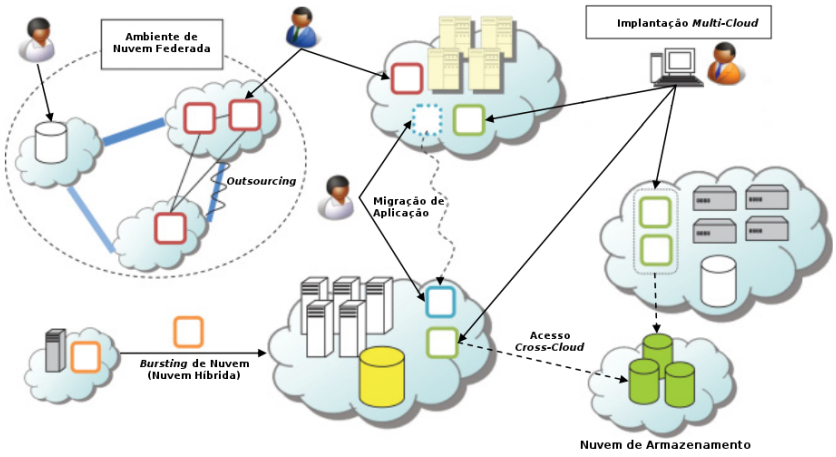

Figura 3.2: Tipos de Interações entre Nuvens, Adaptado de [11].

Segundo Goiri *et al* [47], como é necessário que haja cooperação entre duas ou mais nuvens diferentes em uma federação de nuvens, é possível notar dois comportamentos distintos, os quais são vistos na Figura 3.2, e descritos a seguir:

- ***Outsourcing:*** no momento em que o provedor percebe que seus recursos estão se esgotando, ele busca recursos de outros provedores, para que a qualidade de seu serviço não seja comprometida;
- ***Insourcing:*** este comportamento se dá quando o provedor possui recursos subutilizados, fornecendo-os para provedores externos que necessite-os. Assim, o provedor com recursos sobrando pode economizar um consumo excessivo de sua infraestrutura.

3.5.1 Estágios da Nuvem Federada

De acordo com Celesti *et al.* [12], a evolução do mercado de computação em nuvem pode ser teorizada em três etapas:

- **Monolítico:** são grandes ilhas proprietárias, com serviços fornecidos por empresas de grande porte, como Google[1], Amazon[2] e Microsoft[3], de forma independente umas das outras, conforme é apresentado na Figura 3.3;

[1]Google Cloud Platform, https://cloud.google.com/
[2]Amazon Web Services, https://aws.amazon.com/
[3]Microsoft Azure, https://azure.microsoft.com/

Figura 3.3: Estágio Monolítico [10].

- **Cadeia Vertical de Suprimentos:** possui foco em grandes ilhas proprietárias, assim como o modelo monolítico, mas as empresas começam a utilizar serviços específicos de provedores de nuvens menores, integrando-as ao mercado, de acordo com a Figura 3.4;

Figura 3.4: Estágio Vertical [10].

- **Federação Horizontal:** vários provedores de nuvens de diversos tamanhos fazem acordos entre si, para obterem maiores ganhos de economia de escala e buscando eficiência de utilização de seus recursos e apliação do poder computacional, conforme ilustrado na Figura 3.5, formando uma nuvem-de-nuvens [11, 14].

Figura 3.5: Estágio Horizontal [10].

3.5.2 Arquitetura de Federação de Nuvens

Existem na literatura várias propostas de arquitetura de plataforma de federação em nuvem [12, 13, 17], ou seja, não há um único padrão de arquitetura. Assim sendo, nesta seção serão apresentadas duas propostas de arquitetura, e a plataforma escolhida para ser utilizada neste trabalho, chamada BioNimbuZ [17], que será detalhada no próximo capítulo.

Para a federação de nuvem, Celesti *et al.* [12] propõem uma arquitetura que será detalhada a seguir e representada na Figura 3.6. Em princípio este modelo categoriza as nuvens em dois grupos, local e estrangeiro. A nuvem local trata-se de um provedor que está atendendo o usuário e atinge o limite do poder computacional, precisando então repassar demandas para o restante da federação. Os provedores que receberão as demandas chamam-se estrangeiros e podem cobrar ou não por fornecerem recursos ociosos para atender as demandas recebidas pela nuvem local.

É importante destacar que uma nuvem pode ser local e estrangeira ao mesmo tempo, pois podem ocorrer casos no qual a nuvem local, ao realizar as demandas do usuário, estejam com determinado recurso no seu limite, enquanto outro recurso esteja ocioso e disponível para outro provedor. Este repasse é realizado de forma que atenda os contratos de serviço [4].

Na arquitetura proposta, há um gerenciador chamado de *Cross-Cloud Federation Manager* (CCFM), correspondente a cada provedor na federação. Cada CCFM tem a função de gerenciar sua respectiva nuvem, verificando seu poder computacional e, se necessário, buscando recursos ociosos em outras nuvens. O CCFM é dividido em três subcomponentes:

- ***Discovery Agent:*** este agente possui a responsabilidade de descobrir outras nuvens dentro da federação. Para que isso ocorra, cada provedor precisa disponibilizar suas informações centralizados em um único local, para que possa ser consultado;
- ***Match-Making Agent:*** agente responsável por selecionar a nuvem estrangeira que mais se adequa às demandas necessárias. Também possui a responsabilidade de assegurar que os acordos de níveis de serviço não sejam violados, e que a qualidade de serviço seja mantida;
- ***Authentication Agent:*** este agente é responsável pela autenticação do usuário nas nuvens estrangeiras. Nesta tarefa é possível encontrar vários desafios, como lidar com as diferentes tecnologias de autenticação. Este agente possui a responsabilidade de realizar a interoperabilidade entre essas tecnologias.

Uma outra proposta foi apresentada por Buyya *et al.* [13], na qual afirma-se que o usuário deve utilizar um componente externo aos provedores para interagir com a federação, chamado *Cloud Broker* (CB). O CB possui a responsabilidade de realizar uma intermediação entre o usuário e o provedor, identificar quais provedores possuem recursos disponíveis, mantendo a qualidade de serviço e a não-violação do SLA (*Service Level Agreement* ou acordo de nível de serviço [4]).

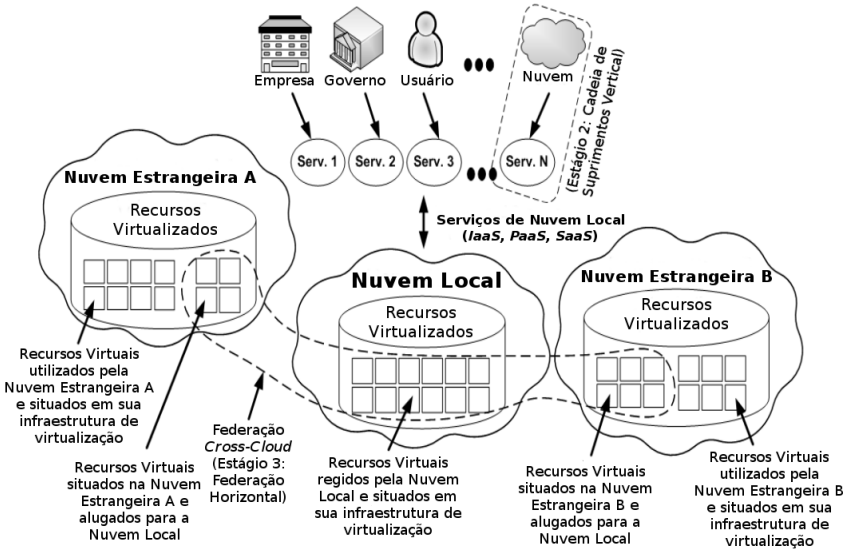

Figura 3.6: Arquitetura proposta por Celesti*et al.* [12], adaptado de [4].

O CB utiliza o *Cloud Exchange* (CEx), um outro componente da arquitetura, para realizar consultas e obter informações sobre padrões de execução, custo de utilização, recursos disponíveis e mapear requisições de usuários aos provedores.

Outro componente desta arquitetura é o *Cloud Coordinator* (CC), que encontra-se dentro de cada provedor na federação, e é responsável por apresentar as informações do provedor para os interessados, e incluir a infraestrutura disponível na federação. No caso de uma demanda de um usuário, o CC estabelece um acordo de qualidade de serviço com cada CB. Esta arquitetura é apresentada na Figura 3.7.

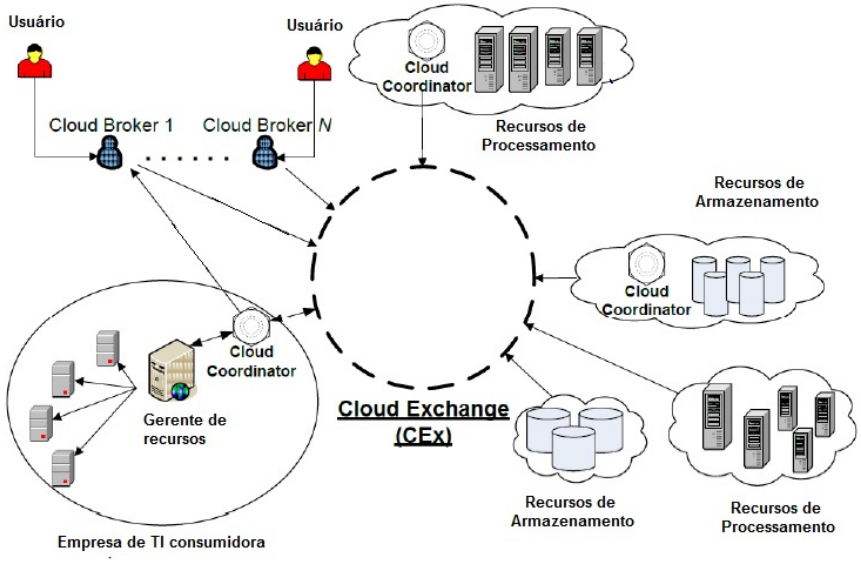

Figura 3.7: Arquitetura proposta por Buyya [13], adaptado de [4].

3.5.3 Vantagens da Federação em Nuvem

Existem muitas vantagens no uso de computação em nuvem. Entre elas, é possível destacar os benefícios a seguir [12, 14]:

- **Melhor aproveitamento de recursos:** os recursos que não estão sendo utilizados podem ser alugados para outros provedores, reduzindo custos de manutenção;
- **Questões legais:** caso haja alguma restrição com os dados ou sistemas a serem utilizados, eles podem ser implantados em regiões geográficas que possam ser adequadas legalmente;
- **Compartilhamento:** os provedores, mesmo com políticas organizacionais diferentes e capacidade técnica distintas, compartilham informações entre si;
- **Tolerância a falhas:** como os serviços são replicados em diversos provedores, mesmo com a interrupção de um deles, o serviço se mantem disponível;
- **Melhor nível de qualidade de serviço:** oferece mais opções para diminuir a latência e atrasos relacionados ao tempos de resposta, devido a utilização de provedores mais próximos geograficamente ou mais capacitados;
- **Otimização de custos:** possibilidade de reduzir custos ao provisionar recursos mais baratos;
- **Redução na violação de SLA:** ao escalonar recursos, um provedor pode alugar ou alocar recursos de outro provedor da federação, para evitar uma possível violação na SLA;
- **Indepêndencia do provedor:** o usuário pode usar qual provedor preferir, sem haver depêndencia de um único.

3.5.4 Desafios da Federação de Nuvens

A federação de nuvem retirou várias limitações que as nuvens computacionais possuíam. No entanto, ainda existem diversas dificuldades e desafios, pois uma federação depende de vários fatores. Assim sendo, os problemas mais comuns são [10, 11, 14]:

- **Carência de padrões abertos:** cada provedor possui seu próprio padrão (por exemplo, mecanismos de segurança e autenticação, comunicação, máquinas virtuais e armazenamento), causando dificuldades ao desenvolver qualquer sistema que apresente soluções para nuvens. Por isso, ocorre o problema *vendor lock-in*, situação na qual as empresas ficam presas à tecnologias, pois a migração de negócios para outras tecnologias são onerosas e com alto custo;

- **Manutenção de SLA:** cada provedor de nuvem possui um SLA (Acordo de Nível de Serviço ou *Service Level Agreement*) diferente, proprorcionando dificuldades para garantir que cada SLA esteja em comum acordo. Enquanto grandes provedoras cobrem custos em relação à violação de acordos para que não haja perda de clientes, pequenas empresas não conseguem agir da mesma forma, pois tais custos prejudicariam seus negócios;
- **Dificuldade de monitoramento:** com a possibilidade de uma federação de nuvens possuir uma grande quantidade de nuvens e estarem em diversas regiões diferentes, geograficamente distantes, há a dificuldade em monitorar os recursos da federação;
- **Confiança:** as empresas provedoras têm resistência em manter suas informações fora de seu domínio, por questões de segurança e de controle.

3.6 Considerações Finais

Neste capítulo foram apresentados os conceitos de computação em nuvem, suas características principais, seus modelos de serviço e seus modelos de implantação. Além disso, foi descrita uma visão geral de federação de nuvem. Tratando-se de federação de nuvem, foram discutidos as formas de operação possíveis, as vantagens deste modelo e os seus desafios. Além disso, afirma-se na literatura que não há um padrão de arquitetura de nuvem federada. Por isso, foram apresentados dois modelos de arquitetura de nuvem federada, uma proposta por Celesti *et al.* [12] e outra proposta por Buyya *et al.* [13].

Para este trabalho, a plataforma de nuvem federada selecionada foi o BioNimbuZ [17], que será descrita em detalhes no próximo capítulo. Ela foi escolhida por apresentar facilidade na execução, e pela possibilidade de ser usada para executar *workflows* de diferentes áreas.

Capítulo 4

Plataforma BioNimbuZ

Este capítulo apresenta a plataforma de nuvens federadas híbridas BioNimbuZ e sua arquitetura atual, proposta por Lopes [10]. Inicialmente, a Seção 4.1 aborda o histórico da plataforma, desde a arquitetura original proposta por Saldanha [17] e, posteriores, melhorias realizadas. Em seguida, a Seção 4.2 apresenta a arquitetura do BioNimbuZ 2, a versão atual da plataforma, que será detalhada nas Seções 4.3, 4.4, 4.5, 4.6. Por fim, a Seção 4.7 aborda o fluxo de execução de um *workflow* no BioNimbuZ.

4.1 Visão Geral

O BioNimbuZ, inicialmente, foi chamado de BioNimbuS. Isso ocorreu na sua primeira versão proposta por Saldanha [17]. O objetivo do BioNimbuS é permitir a integração e o controle de diferentes provedores de computação em nuvem, públicas e privadas [17]. Além do trabalho de Saldanha, o BioNimbuZ passou por várias melhorias [3], [4], [19], [20], [21]. Recentemente, o BioNimbuZ passou por uma nova reformulação, e desta vez foi realizada por Souza [10] e Gomes [14], dando origem ao BioNimbuZ 2.

Assim, o BioNimbuS passou a ser chamado de ZooNimbuZ [19] em 2013, quando Moura e Barcelar implementaram a integração do BioNimbuS com as ferramentas Apache Zookeeper [48] e Apache Avro [49]. No BioNimbuS, a comunicação entre as camadas de núcleo e de infraestrutura eram realizadas por meio de P2P (*peer-to-peer*). Na versão ZooNimbuS a comunicação passou a ser realizada via *Remote Procedure Call* (RPC), fornecendo escalabilidade e flexibilidade por meio do Apache Avro. O Apache Zookeeper, por sua vez, ofereceu uma centralização de dados na comunicação entre servidor e clientes, garantindo mais confiabilidade. No entanto, no final de 2013, o ZoonimbuS passava a se chamar BioNimbuZ [50], de acordo com a nomenclatura atual.

Posteriormente, foi implementado por meio de Azevedo e Freitas Junior [51] uma política que considerasse a quebra de arquivos e a largura da banda entre cliente e servidor

para transferências e compactação de arquivos. Depois, Ramos [3] implementou uma interface gráfica para o sistema e um controlador de *jobs* para ligar a interface com o núcleo do agora denominado BioNimbuZ [20]. Nessa nova versão, ainda foi implementado por Vergara [21] um controlador de elasticidade, permitindo a flexibilização de acordo com os requisitos das máquinas virtuais utilizadas pelo usuário e pela plataforma.

O BioNimuZ é considerado um sistema gerenciador de *workflow* e, assim como ele, existem outros sistemas no mercado. Entre algumas opções, se destaca o sistema Pegasus [52], uma plataforma desenvolvida pela USC(*University of Southern California*), que também utiliza o conceito de computação em nuvem para a execução de *workflows* científicos. Esta plataforma, assim como o BioNimbuZ, possui características como escalabilidade, tratamento de erros e confiabilidade. Além disso, ela é capaz de executar *workflows* de áreas como astronomia, bioinformática, ciência de terremotos e de oceanos, entre outros. Outro sistema que surgiu recentemente é o Apache Airflow [53], que permite criar, monitorar e executar *workflows* em nuvem ou localmente, por meio de DAGs (*directed acyclic graphs* ou grafo acíclico dirigido) de tarefas, representado por um *script*.

4.2 Arquitetura do BioNimbuZ

A atual versão do BioNimbuZ, chamada de BioNimbuZ 2, foi desenvolvida por Lopes [10] e Gomes [14], possui uma arquitetura mais modularizada, com camadas bem divididas, facilitando a integração de seus componentes na federação de nuvens.

A nova arquitetura tem como característica a utilização de *plugins* de microsserviço independentes, podendo ser criados ou encerrados a qualquer momento [54]. Além disso, a comunicação entre os componentes pode ser realizada em alto nível, com transmissões de informações de forma mais simples, além de ser realizada entre as nuvens, mas também é realizada em nível mais baixo, feita dentro da nuvem, possibilitando que as informações se atualizem em tempo real. Também é possível cadastrar as credenciais de nuvens dos usuários dentro da plataforma BioNimbuZ para utilização de seus recursos, além da implementação de uma funcionalidade que permite o compartilhamento com diversos grupos de usuários (como o grupo da instituição de ensino, por exemplo) [14].

A arquitetura da segunda versão do BioNimbuZ é composta por quatro camadas [10]: Camada de Aplicação, Camada de Federação, Camada de Coordenação e Camada de Execução, conforme apresentado na Figura 4.1, e detalhadas a seguir nas Seções 4.3, 4.4, 4.5, 4.6, respectivamente.

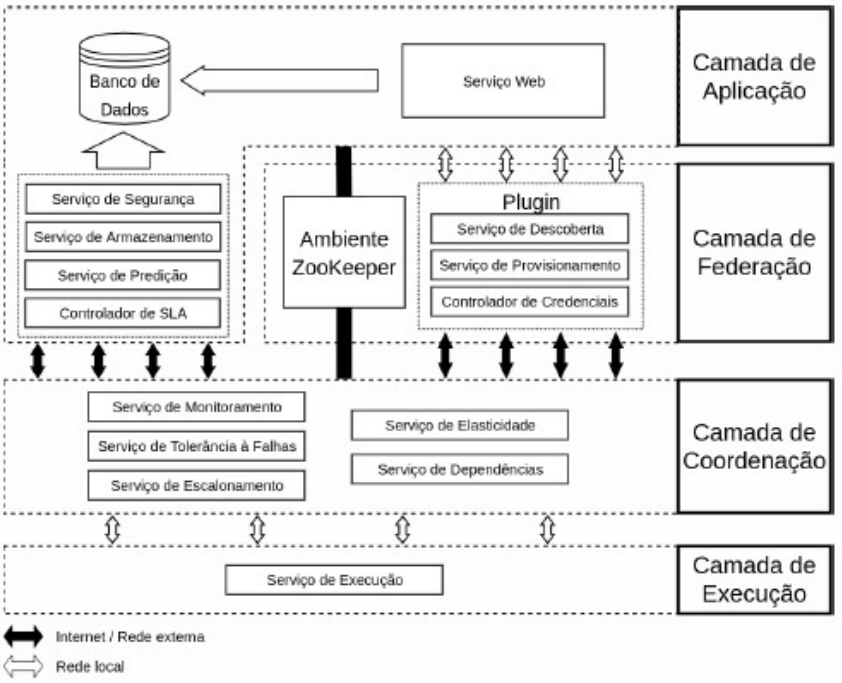

Figura 4.1: Arquitetura da Nova Versão do BioNimbuZ [14].

4.3 Camada de Aplicação

É a camada que interage diretamente com os usuários por meio de uma interface web. Essa interface possui funcionalidades como registrar o usuário, gerenciar suas credenciais de provedores de nuvem, executarem aplicações (as instâncias de um *workflow* científico, por exemplo) e realizar *upload*/*download* de dados para as nuvens. Todos esses dados são inseridos e mantidos em bancos de dados [10]. Os serviços e os controladores desta camada são [10]:

- **Serviço Web:** serviço disponível para apresentar a interação dos usuários com a plataforma por meio de páginas web. Assim, os usuários podem acessar a plataforma BioNimbuZ, desenhar *workflows*, realizar *uploads* ou *downloads*, e monitorar os *workflows* que estão executando, entre outras funcionalidades;
- **Serviço de Segurança:** este serviço possui a responsabilidade de garantir a integridade e a confidencialidade de informações dos usuários, como dados de autenticação do usuário, credenciais de serviço de nuvem ou *workflows*, mantendo-os de forma sigilosa e protegida de pessoas sem permissão de acessá-las ou alterá-las[14];
- **Serviço de Armazenamento:** este serviço é responsável por escolher a melhor decisão a respeito do armazenamento dos arquivos de entrada e de saída das tarefas

executadas, verificando o custo, a localização e a capacidade. Isso é uma estratégica semelhante à de armazenamento de arquivos fornecida por provedores de nuvem;

- **Serviço de Predição:** serviço que auxilia o usuário a definir os recursos para que o *workflow* seja executado, no quesito tempo, custo e recursos computacionais, após o usuário apresentar o *workflow* que será executado. As plataformas de nuvem que serão consideradas na tomada de decisão do serviço serão apenas as plataformas na qual o usuário apresentou as credenciais, e não todas as plataformas em nuvem que a plataforma suporta;
- **Controlador de SLA:** este serviço controla os modelos de SLAs configuradas pelos usuários para a execução de tarefas, e garante que os acordos não sejam violados. Tais acordos são constituidos de requisitos por parte do usuário, tal como quantidade de núcleos do processador, memória e armazenamento solicitado. Assim, o controlador confere com o provedor da nuvem se tais solicitações podem ser atendidas.

4.3.1 Plataforma Web do BioNimbuZ

Com o intuito de apresentar a interface gráfica, os *menus* e as opções da segunda versão do BioNimbuZ, esta subseção é apresentada.

Para realizar o *login* e entrar no sistema, conforme apresentado na Figura 4.2, há duas permissões de acesso, ou seja, dois tipos de usuários: o Administrador e o Cliente. O perfil de Cliente é mais básico, voltado para o uso da plataforma, enquanto o perfil de Administrador foca na gerência, além de acumular as funcionalidades disponíveis para o perfil de Cliente.

Quando o usuário realiza o *login* como Administrador, é apresentada a tela principal, com mais opções, conforme mostrado na Figura 4.3. Na tela, são apresentadas as seguintes opções:

- ***Plugins*:** nesta opção há mais dois subitens. O item ***Administration***, para adicionar *plugins* de nuvens, e o item ***Price Tables*** para consultar o *status* da tabela de preços dos *plugins*;
- ***Credentials:*** neste item o usuário pode fornecer suas credenciais de nuvem que serão utilizadas para a criação e a execução das tarefas e dos *workflows*;
- ***Groups:*** neste item os usuários podem configurar os grupos de usuários, permitindo o compartilhamento de credenciais e da memória para armazenamento;
- ***Images:*** neste item é possível buscar e selecionar imagens de sistemas operacionais das plataformas de nuvem;

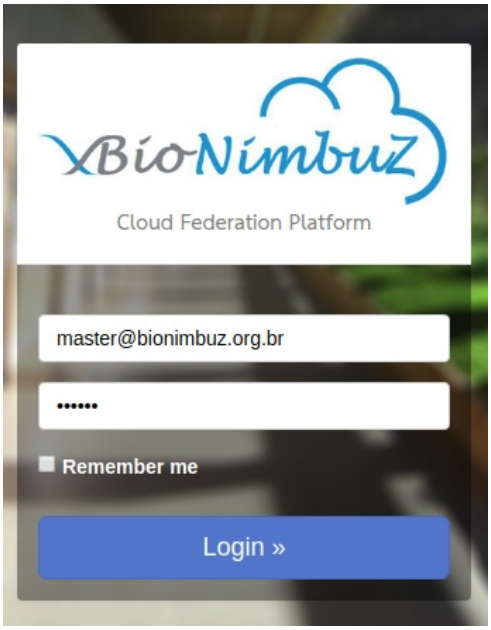

Figura 4.2: Tela de Login da Segunda Versão da Plataforma BioNimbuz [10].

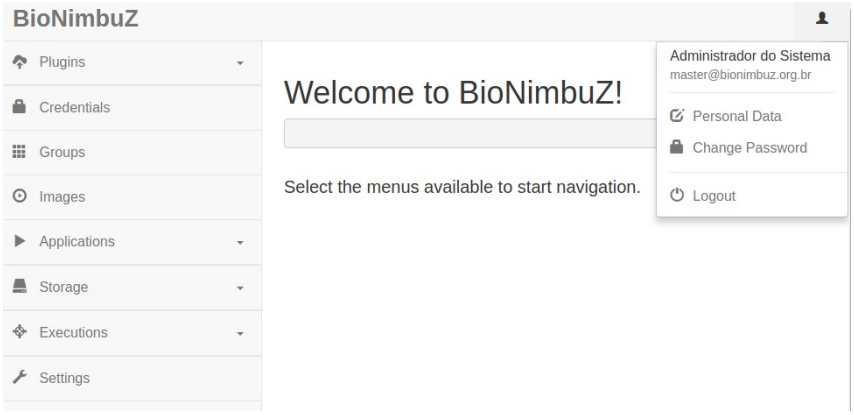

Figura 4.3: Tela inicial do BioNimbuz para Administrador [10].

- ***Applications:*** nesse item é possível gerenciar as tarefas que os usuários executarão, por meio dos *scripts* de inicialização e de execução. Esta opção possui dois subitens: o item ***Coordinator*** para configurar um *script* padronizado, para a coordenação das tarefas que estão sendo executadas, e o item ***Executors*** para que o usuário, com o perfil de Administrador, possa adicionar tarefas na plataforma, permitindo assim que o usuário com perfil de Cliente possa executá-las;
- ***Storage:*** este item é relacionado ao armazenamento de arquivos ao acessar um dos dois subitens. O item ***Spaces*** possui a funcionalidade para criar espaços de armazenamento na nuvem de preferência do usuário, e o item ***Files*** permite que o usuário realize *upload/download* de arquivos em um dos espaços existentes;

- ***Executions:*** contém a principal funcionalidade da plataforma, que é a execução de tarefas, por meio de dois subitens. O item ***Instances***, que permite executar uma instância por meio de uma máquina virtual, na nuvem selecionada pelo usuário, ou o item ***Workflows***, no qual é possível montar um *workflow* por meio de interface gráfica, criando o fluxo de execução das tarefas, formando um conjunto de instâncias e de máquinas virtuais, e montando as depêndencias entre elas.

4.4 Camada de Federação

Segundo Lopes [10], a Camada de Federação pode ser definida como "responsável por garantir a integração dos diferentes componentes do sistema que podem estar situados em diversas nuvens, formando uma federação de nuvens". Isso é possível devido ao Apache ZooKeeper [48], que mantem os dados distribuídos entre a Camada de Coordenação e a Camada de Aplicação. Esses dados vão desde estado de execução de tarefas até endereços das máquinas monitoradas pelos coordenadores.

Esta camada também possui *plugins*, implementados com a abordagem de microsserviços [54], no qual cada um possui uma pequena responsabilidade e roda em seu respectivo processo, comunicando-se entre si por meio de uma API (*Application Programming Interface*) sobre o protocolo HTTP[1].

Assim, a Camada de Federação possui *plugins* que fornecem os seguintes serviços e controladores:

- **Serviço de Descoberta:** este serviço é responsável por compilar informações de latência de rede, preços e recursos computacionais, e outros serviços de determinada plataforma de nuvem. Essas informações permanecem em memória, para rápido acesso do usuário[14];
- **Serviço de Provisionamento:** este serviço se comunica com a Camada de Aplicação, apresentando a possibilidade de alocar ou desalocar máquinas de determinado provedor. Além disso, possui a responsabilidade de reservar espaço para possíveis *downloads/uploads* de arquivos de entrada e de saída, das tarefas definidas na Camada de Aplicação;
- **Controlador de Credenciais:** este serviço possui a responsabilidade de reconhecer e interpretar credenciais enviadas pela Camada de Aplicação, e traduzí-las para o mecanismo de segurança utilizado pela nuvem.

[1]Protocolo HTTP,https://www.w3.org/Protocols/

4.5 Camada de Coordenação

Esta camada e seus serviços são formados por Coordenadores de Tarefas (*Task Coordinator* - TC), que possuem a responsabilidade de monitorar e garantir o fluxo das tarefas criadas pelos usuários, além das dependências entre elas.

O funcionamento inicia no momento em que as tarefas são criadas e a sua execução é solicitada. Em seguida, um TC é criado na nuvem de destino, com o intuito de criar uma proximidade maior com a tarefa que será realizado o monitoramento, proporcionando mais eficiência na comunicação.

O Coordenador de Tarefas é criado por meio das credenciais do usuário, ou das credenciais do grupo de usuários, e como consequência disso, é incluído para o usuário um custo extra devido à execução de Coordenadores da plataforma em nuvem, além do custo total para que uma tarefa seja executada (como alocação de máquinas virtuais e armazenamento). Esse custo adicional oferece ao usuário benefícios como provisionamento e tolerância a falhas, devido ao uso da plataforma de federação.Assim, os serviços oferecidos pela Camada de Coordenação são [10]:

- **Serviço de Monitoramento:** este serviço possui a responsabilidade de monitorar os recursos computacionais utilizados pelo Coordenador, além do consumo do Executor de Tarefas (*Task Executor - TE*, a ser detalhado na Seção 4.6). Esse monitoramento não é direcionado para a cobrança de tarifas para o usuário, mas para a garantia do SLA. Por isso, esse serviço possui contato constante com TEs, coletando dados de execução;

- **Serviço de Dependências:** este serviço possui a função de verificar as demandas das tarefas, para que possa monitorar suas dependências. Isso significa que esse serviço verifica quais serão as próximas tarefas a serem executadaas, ou então se o TC pode ser encerrado para economizar gastos;

- **Serviço de Tolerância a Falhas:** este serviço trabalha junto ao Serviço de Monitoramento. Quando o Serviço de Monitoramento detecta o encerramento das tarefas e realiza a replicação dos resultados gerados, garantindo que haja ao menos duas cópias. Assim, quando se nota que algum resultado não possui replicação disponível, a tolerância a falhas dá início ao processo de duplicação;

- **Serviço de Elasticidade:** este serviço trabalha com auxílio do Serviço de Monitoramento, coletando informações de execução de tarefas dos TEs, para que possa decidir quando deve aumentar ou dimunuir recursos computacionais (como quantidade de máquinas virtuais tamanho de memória, etc);

- **Serviço de Escalonamento:** reponsável por distribuir as tarefas que devem ser executadas na nuvem em que o TC está executando, e solicitar o início da execução das tarefas.

4.6 Camada de Execução

Esta camada é composto por Executores de Tarefas (*Task Executor* - TE), que são processos que fornecem o Serviço de Execução, que é responsável por manter o ciclo de vida das tarefas e o seu monitoramento.

Devido a esse monitoramento, o TE se comunica constantemente com os TCs, enviando consumo de processador, de memória ou estado de execução da tarefa. Por isso, esta camada é responsável por executar e monitorar as tarefas que compõem o *workflow* formado na Camada de Aplicação.

Além do Serviço de Execução, a Camada de Execução oferece o Serviço de Obtenção de Recursos, responsável por obter os arquivos de entrada que serão requisitados durante a execução da tarefa; e o Serviço de Persistência, que é iniciado para realizar o *upload* dos arquivos de saída para a nuvem especificada no momento da criação da tarefa.

4.7 Execução de *Workflows*

Esta seção tem como objetivo apresentar o fluxo principal de execução do BioNimbuZ, que representa as tarefas de adicionar o arquivo de entrada e executar o *workflow*, ou uma simples tarefa.

Considerando que o usuário já apresentou as credenciais de plataforma de nuvens no BioNimbuZ, ele deve, a partir da tela inicial do sistema, selecionar a opção *Space*, para criar um espaço de armazenamento. Na tela apresentada na Figura 4.4, ele deve selecionar a opção *Add Space* para criar o espaço, caso ele ainda não esteja criado. Assim, será possível criar o nome do espaço de armazenamento e o local no qual ele será armazenado.

Após a criação do espaço, o próximo passo será realizar o *upload* dos arquivos de entrada. Para isso, o usuário deve selecionar a opção *Files*, dentro do menu *Storage*. Para adicionar o arquivo de entrada, o usuário deve selecionar a opção "*Add Space File*", onde será apresentada a tela para a realização do *upload*, conforme mostrado na Figura 4.5. Agora, o usuário deve selecionar o espaço de armazenamento e o arquivo que será adicionado. Ainda é possível alterar o nome do arquivo, de forma optativa.

Em seguida, o próximo passo é a criação e a execução do *workflow*. Se o usuário desejar realizar a execução de apenas uma tarefa, ele tem a opção de selecionar a opção *Instances*, dentro do menu *Executions*. Dessa forma, será apresentada a lista de tarefas

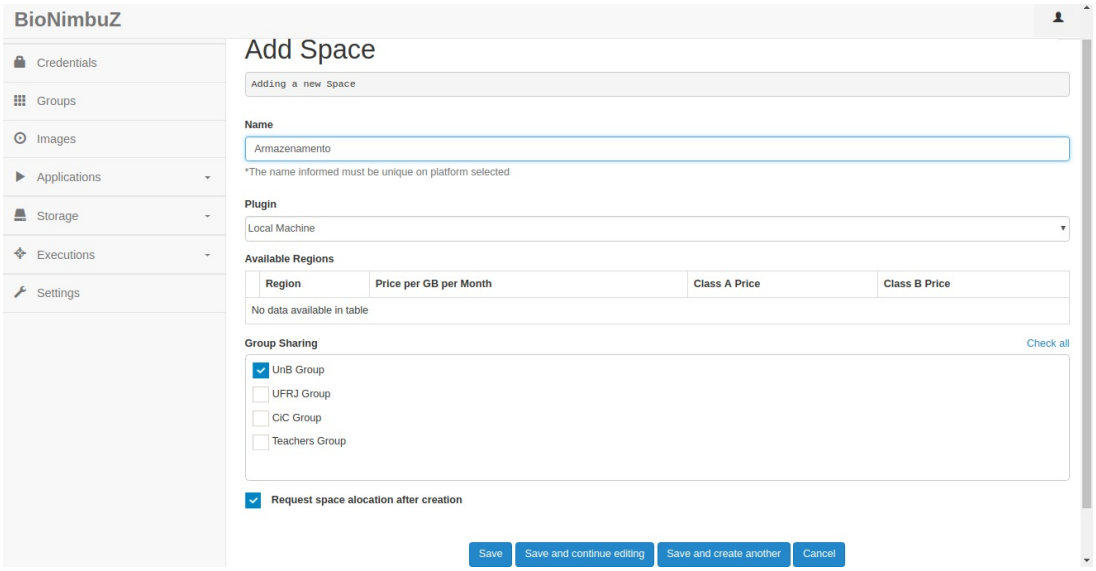

Figura 4.4: Tela para Criar um Novo Espaço.

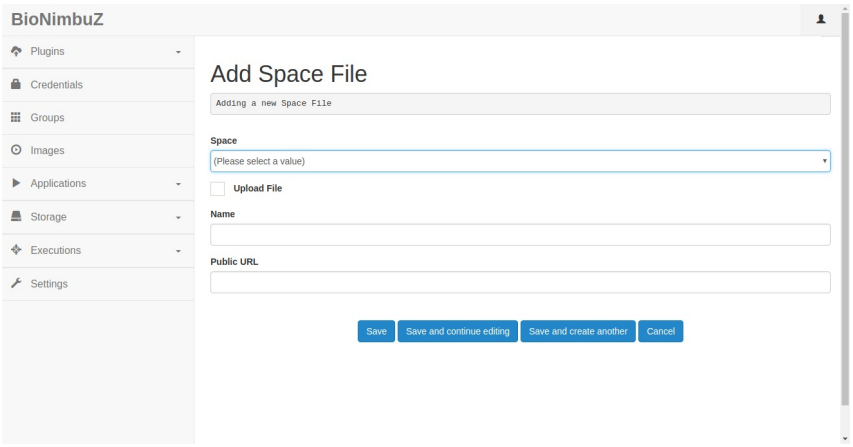

Figura 4.5: Tela para Realização de *Upload* de Arquivo.

executadas e seus respectivos *status*, para acompanhamento de sua execução. No entanto, para criar uma nova tarefa, a opção *Add Stance* deve ser selecionada. Na tela que será apresentada, o usuário seleciona a tarefa que deseja executar, através do menu *Executor* e, após a seleção, é apresentado na tela as opções para sua criação, conforme apresentado na Figura 4.6.

Caso o *workflow* possua mais de uma tarefa, o usuário deve selecionar a opção *workflows*, no qual é apresentada a lista de *workflows* que já foram executados, caso haja. Para criar e executar o novo *workflow*, a opção *Add Workflow* deve ser selecionada. Na tela, é apresentado o diagrama do *workflow*, de acordo com a Figura 4.7, no qual é possível adicionar as tarefas e editá-las, conforme Figura 4.6. No diagrama, conforme vão sendo inseridas tarefas, é possível organizar as dependências entre elas por meio de setas, indicando qual será a ordem de execução. Por fim, basta selecionar a opção *Execute* para que

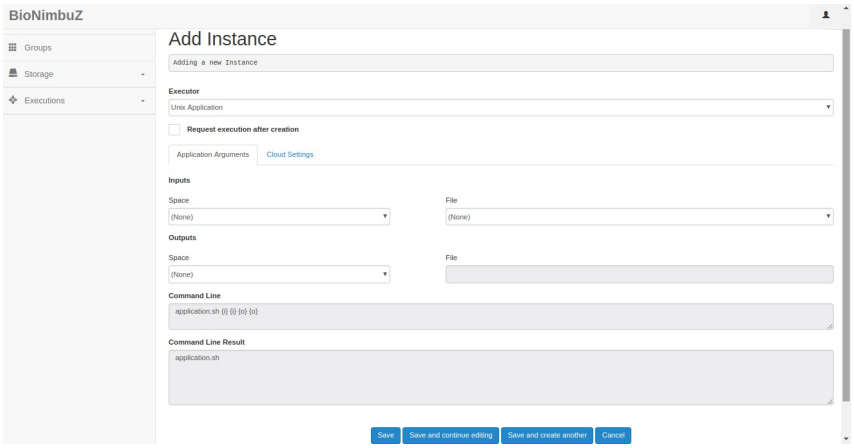

Figura 4.6: Tela de Criação de Tarefa.

o *workflow* seja executado.

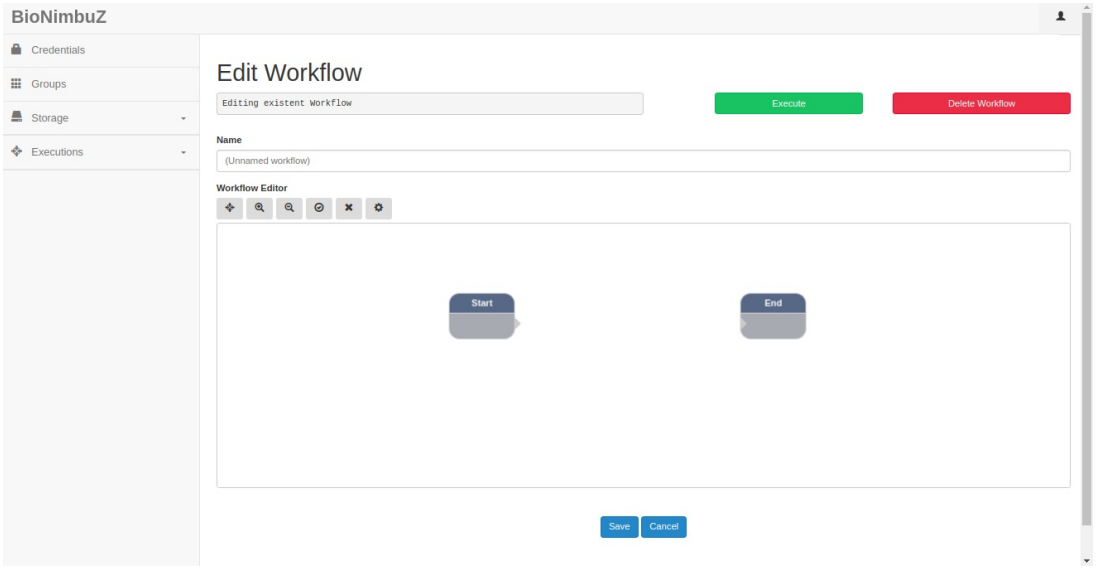

Figura 4.7: Tela de Diagrama do *Workflow*.

A execução de um *workflow* se dá pelo processo de criação de uma instância com os recursos computacionais que foram definidos no momento de implantação da tarefa no BioNimbuZ. Dentro da instância, é realizada a execução de dois *scripts*, o primeiro sendo o *Startup Script*, que instala as bibliotecas necessárias para a execução da tarefa, e faz o *download* dos arquivos de entrada que se encontram no *Space* da plataforma. Logo após, o *Execution Script* é chamado, executando a tarefa em questão, utilizando os arquivos de entrada. Após a execução, se a tarefa foi realizada com sucesso, os arquivos de saída são direcionados para o *Space* informado no momento da criação da tarefa, apresentado na Figura 4.6. Após a execução da tarefa, a instância é encerrada.

4.8 Considerações Finais

Neste capítulo foi apresentada a plataforma de nuvens federadas híbridas BioNimbuZ, desde a sua origem com Saldanha [17], citando suas melhorias, até a segunda versão da plataforma, proposta por Lopes [10]. Além disso, foi abordada a nova arquitetura do BioNimbuZ e suas camadas, detalhando a forma de operação por meio dos serviços disponíveis em cada.

Explica-se, também, a nova versão da interface da plataforma, abordando cada tipo de perfil de usuário, chamados Administrador e Cliente, e as possibilidades de cada tipo de perfil, além de apresentar um fluxo de execução do sistema, que apresenta como inserir arquivos na plataforma até montar e executar um *workflow*.

No próximo capítulo será apresentada a execução de três *workflows*, da área de Astronomia, Bioquímica e Reconhecimento Facial na plataforma BioNimbuZ, e os seus resultados.

Capítulo 5

Resultados

Este capítulo apresenta os detalhes da execução de três *workflows* científicos. A Seção 5.2 apresenta o *workflow* da área de Astronomia. A Seção 5.3 apresenta o *workflow* da área de Bioquimica, e a Seção 5.4 o *workflow* da área de Reconhecimento Facial. Esses três *workflows* foram executados na plataforma de nuvens federadas híbridas BioNimbuZ, demonstrando assim que essa plataforma é capaz de rodar *workflows* de diversas áreas cientificas.

5.1 Inserção de Tarefas no BioNimbuZ

Primeiramente, deve ser realizado um procedimento em comum para os três *workflows* científicos a serem executados, que é a inserção de suas respectivas tarefas na plataforma BioNimbuZ. Para isso, o usuário com perfil de Administrador deve ir no menu *Applications*, a partir do menu inicial, e selecionar a opção *Executors*, inserindo o *script* da tarefa no campo *Execution Script*, conforme é apresentado na Figura 5.1.

É importante destacar que é necessário realizar um levantamento sobre quais são os requisitos para a execução de determinado *script*. Por exemplo, no caso da implantação do *workflow* de Astronomia, foi instalada a biblioteca Montage por meio do gerenciador de pacotes (apt-get) do sistema operacional Linux[1]. Para execução do *workflow* de Reconhecimento Facial, foi instalada a ferramenta pip[2] (um sistema de gerenciamento de pacotes, originário do PyPI - *Python Package Index*) e a biblioteca OpenCV [9]. Para a instalação de pacotes e bibliotecas adicionais, é necessário apenas seguir para a mesma tela de inserção do *script*, mas os comandos de instalação devem ser adicionados no campo *Startup Script*, conforme apresentado na Figura 5.1.

[1]https://www.linux.org/
[2]https://pypi.org/project/pip/

A Figura 5.1 também é a tela na qual deve ser selecionada as máquinas virtuais a serem utilizadas pela tarefa que está sendo inserida, no campo *Images*.

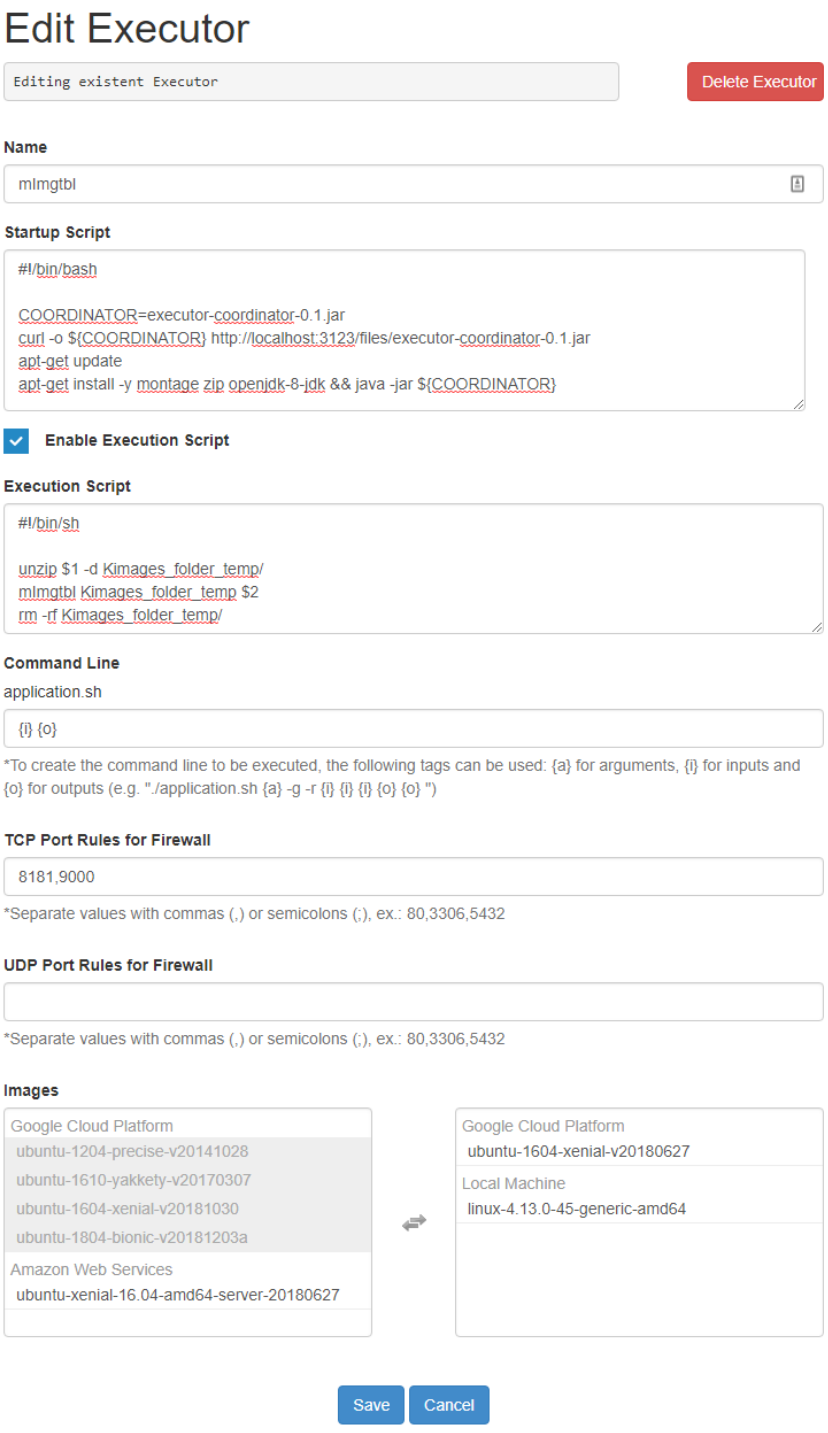

Figura 5.1: Tela de Inserção de Tarefas no BioNimbuZ.

Todas as tarefas dos *workflows* científicos foram executados na nuvem, utilizando máquinas virtuais com os mesmos recursos que são exemplificadas na Figura 5.2. Tal

configuração inclui a escolha da máquina virtual na qual a tarefa foi realizada. O tipo da máquina virtual escolhida é chamada pela Google[3] de *n1-standart-1*, e consiste de 3,75GB de memória RAM e 1 vCPU. Estas máquinas virtuais utilizaram o Sistema Operacional Ubuntu 16.04.

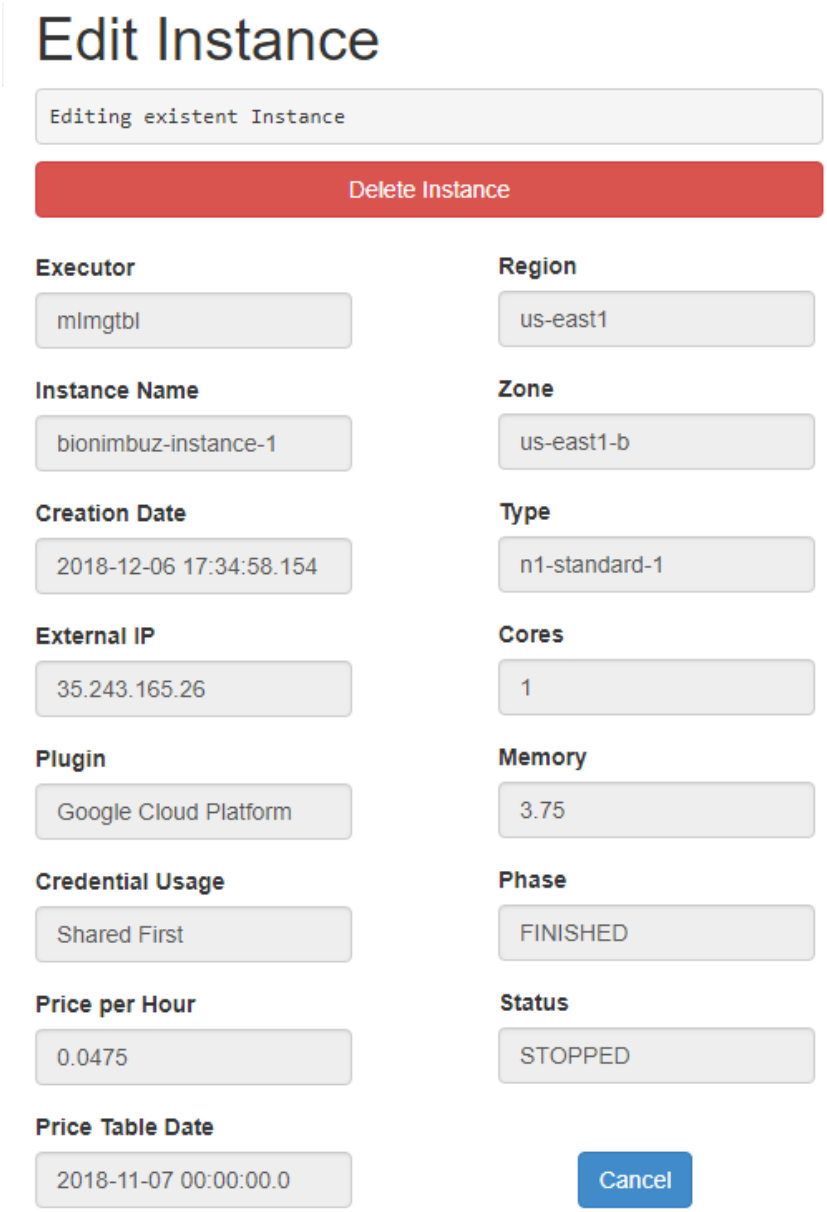

Figura 5.2: Configuração da Tarefa **mImgtbl** no BioNimbuZ.

[3]Google Cloud Platform, https://cloud.google.com/

5.2 *Workflow* de Astronomia

Para a execução do *workflow* de Astronomia foi montado o diagrama apresentado na Figura 5.3, o qual mostra a ordem de execução das tarefas, começando pelo *box Start* e finalizando no *box End*, demonstrando que o fluxo de execução das tarefas é da esquerda para a direita. Além disso, o diagrama apresenta setas, indicando a dependência de cada tarefa e, consequentemente, qual tarefa fornece o arquivo de entrada para a tarefa seguinte. Pelo diagrama é possível ver que em determinado momento ocorre uma Distribuição de Dados na tarefa *mImgtbl*, divindindo em dois fluxos de execução do *workflow*, configurando um paralelismo.

Este *workflow* se caracteriza pela dependência não só de arquivos, mas de diretórios. Assim, determinadas tarefas podem ser realizadas apenas em diretórios como, por exemplo, a tarefa *mAdd*, que requisita um diretório que contém os arquivos FITS como argumento, para realizar a formação do mosaico.

Como a primeira etapa do *workflow* requisita um diretório com 91 arquivos FITS (*Flexible Image Transport System*), fornecidos pela IRSA (*Infrared Science Archive*) [55], foi necessário realizar uma compressão do diretório, pois o BioNimbuZ não permite o *upload/download* de diretórios.

Após a execução de uma tarefa do *workflow*, o *box* do diagrama referente àquela tarefa adquire a coloração verde, indicando que ela foi realizada corretamente. Assim, na Figura 5.5 são apresentadas todas as tarefas do *workflow* de Astronomia com a cor verde, indicando que o *workflow* foi executado com sucesso. As Figuras 5.5 e 5.6 apresentam os dois arquivos de saída gerados por este *workflow*.

5.3 *Workflow* de Bioquímica

Ao executar este *workflow* científico na plataforma BioNimbuZ, é formado o diagrama apresentado na Figura 5.7, representando o desenho do fluxo de execução com três tarefas, em ordem linear de dependência. O arquivo de entrada é fornecido no formato PDB (*Protein Data Bank*), sendo encontrado na plataforma RCSB[4], representando uma proteína. Neste trabalho, foi utilizada a proteína 1t80 [8]. A primeira tarefa realiza uma preparação no arquivo, retirando seu cabeçalho e algumas informações desnecessárias para a análise do conteúdo. Na segunda etapa, é realizada uma retirada de águas cristalográficas da estrutura molecular, para que elas não interfiram na análise do movimento molecular. Por último, é realizada uma verificação de *gaps* das moléculas, procurando por locais que estejam com partes ausentes.

[4]https://rcsb.org/structure/1t80

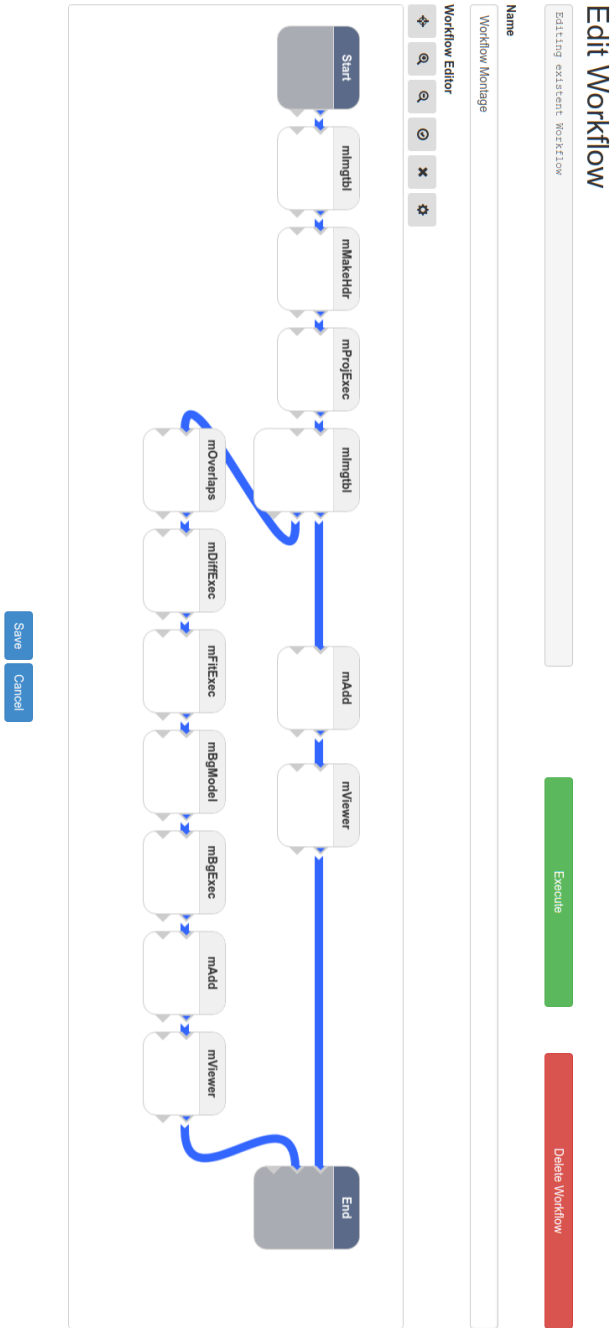

Figura 5.3: Criação do Diagrama de um *Workflow* Científico do Montage no BioNimbuZ.

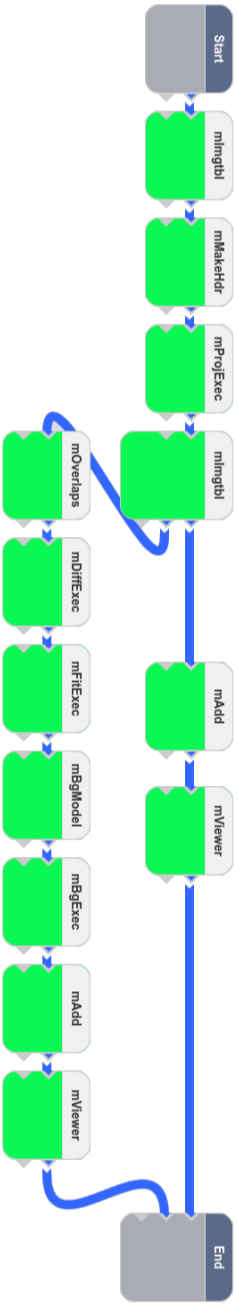

Figura 5.4: Diagrama do *Workflow* Científico do Montage Após Execução no BioNimbuZ.

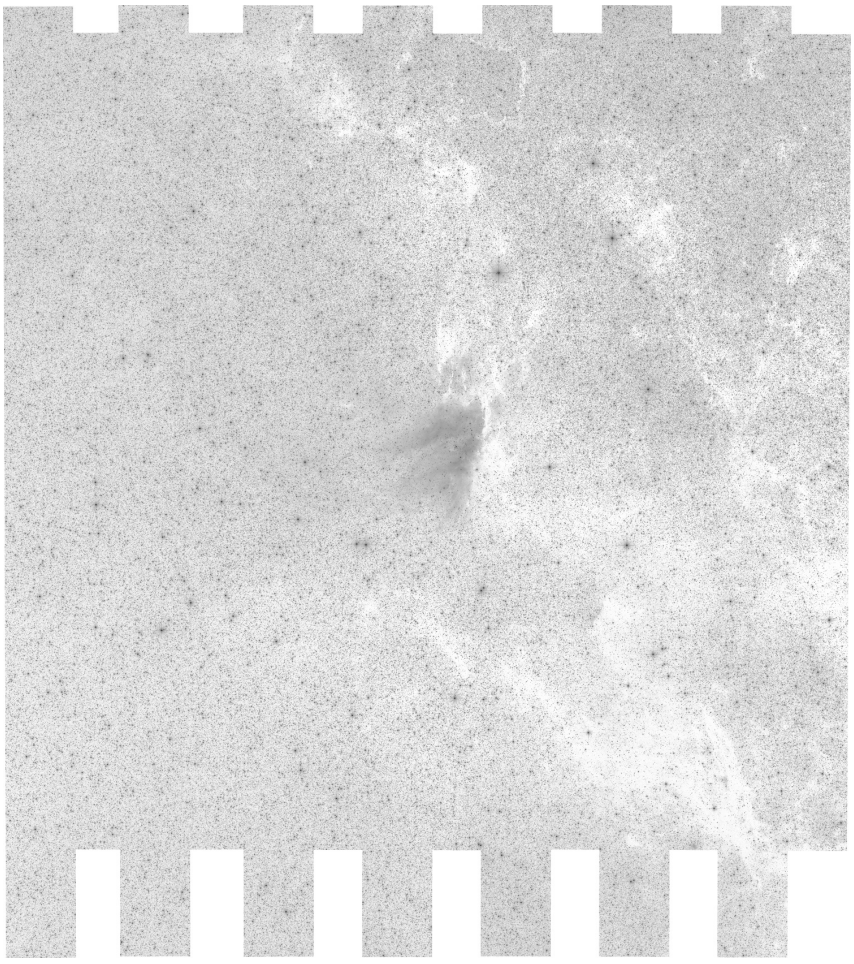

Figura 5.5: Arquivo de Saída do *Workflow* Montage. Imagem com Correções.

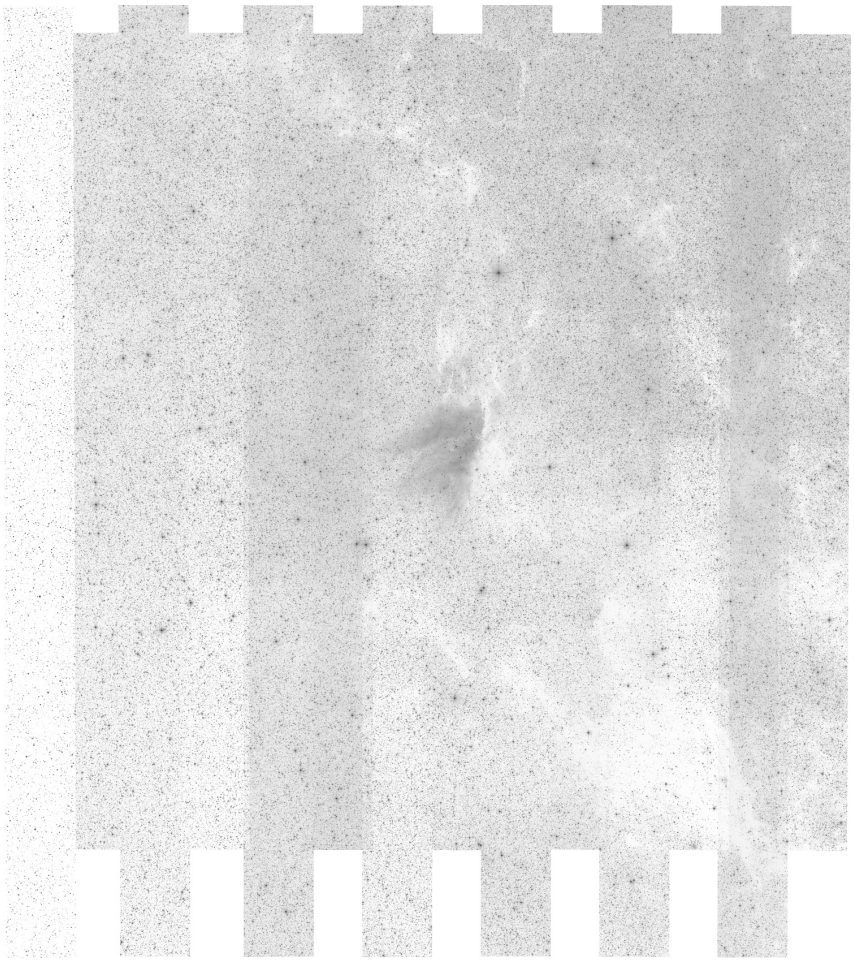

Figura 5.6: Arquivo de Saída do *Workflow* Montage. Imagem sem Correções.

Figura 5.7: Criação do Diagrama de um *Workflow* Científico de Bioquímica no BioNimbuZ.

A execução deste *workflow* foi linear, no qual cada etapa realizou transformações no arquivo de entrada, ou seja, o arquivo de saída possuía o mesmo formato do arquivo de entrada. Assim, o *workflow* científico foi executado com sucesso, conforme é apresentado na Figura 5.8.

Figura 5.8: Diagrama do *Workflow* Científico de Bioquímica após Execução no BioNimbuZ.

5.4 *Workflow* de Reconhecimento Facial

Para a execução do *workflow* de Reconhecimento Facial (ou Visão Computacional), foi montado o diagrama conforme mostrado na Figura 5.9, que apresenta um *workflow* linear, no qual a segunda tarefa necessita de arquivos da primeira. Neste caso, a primeira tarefa utiliza uma imagem fornecida como arquivo de entrada para o *workflow* e realiza um reconhecimento de todas as faces presentes na imagem, armazenando em um diretório. Na segunda tarefa é repassado o diretório originado na primeira tarefa, além do usuário fornecer uma outra imagem com rosto, para comparar com as faces armazenadas no diretório. Assim como no *workflow* de Astronomia, este *workflow* utiliza diretórios como arquivo de saída da primeira tarefa e entrada para a segunda tarefa, necessitando assim realizar compressão do diretório.

Neste *workflow* foram utilizados duas imagens com a mesma face, porém com expressões diferentes. Na Figura 5.11 é mostrado o arquivo de entrada da primeira tarefa, no qual a face da imagem é capturada e armazenada. A segunda tarefa captura a face fornecida como entrada e a compara com a imagem armazenada da primeira tarefa, fornecendo dois arquivos de saída, sendo uma imagem com a identificação de qual amostra foi identificada e comparada, e um arquivo informando a amostra e uma pontuação que representa um grau de confiança, no qual quanto maior, mais diferente são as faces.

Além do desafio de transformar diretórios em arquivos comprimidos, outra dificuldade superada foi em relação à instalação da biblioteca OpenCV [9], conforme detalhado na

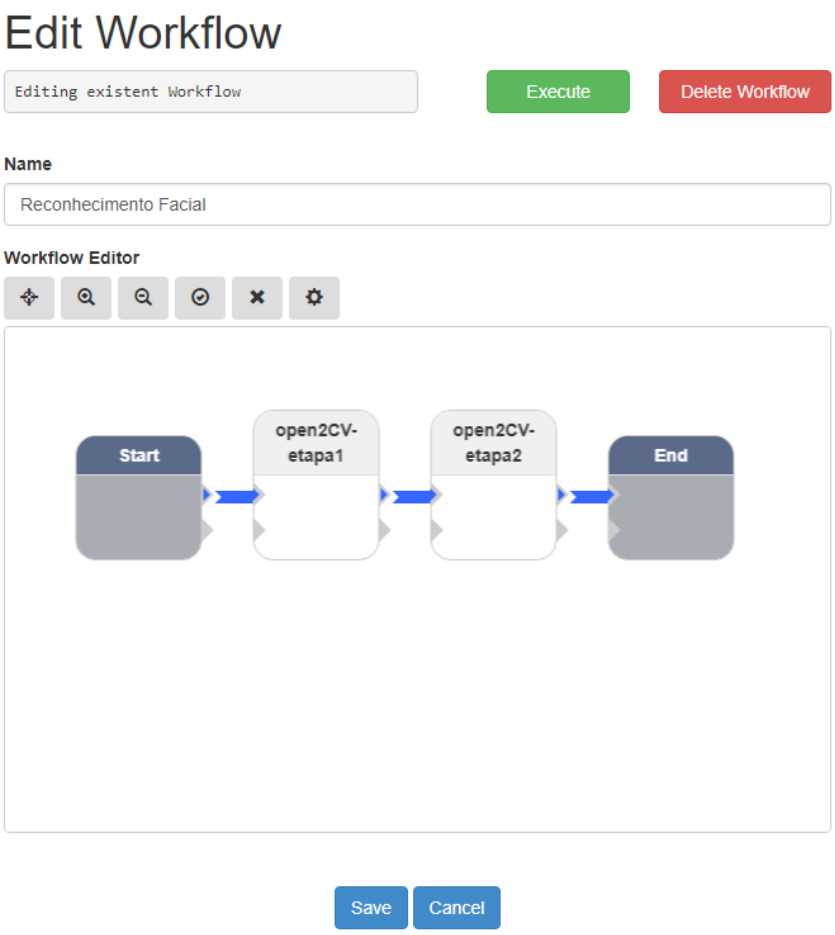

Figura 5.9: Criação do Diagrama de um *Workflow* Científico de Reconhecimento Facial no BioNimbuZ.

Seção 5.1. No entanto, o *workflow* foi executado com sucesso, conforme mostrado na Figura 5.10, apresentando na Figura 5.12 o arquivo de saída e o arquivo de texto.

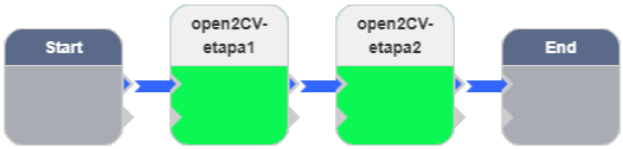

Figura 5.10: Diagrama do *Workflow* Científico de Reconhecimento Facial após Execução no BioNimbuZ.

Figura 5.11: Arquivo de Entrada para a Primeira Tarefa do *Workflow* de Reconhecimento Facial.

5.5 Considerações Finais

Neste capítulo foram apresentados os resultados da execução de três *workflows* científicos das áreas de pesquisa de Astronomia, de Bioquímica e de Reconhecimento Facial, apresentando seus arquivos de saída, destacando os desafios de cada *workflow* para que

suas tarefas possam ser realizadas, além de apresentar algumas características de cada *workflow* científico.

Figura 5.12: Arquivo de Saída do *Workflow* de Reconhecimento Facial.

Capítulo 6

Conclusão e Trabalhos Futuros

O BioNimbuZ supre a demanda de uma plataforma de nuvens federadas que permite executar *workflows* científicos de forma simples, permitindo o usuário gerenciar suas autenticações em diversos provedores, e gerenciando a melhor forma de utilizar os recursos fornecidos pelas nuvens na execuções de tarefas.

Diante deste contexto, este trabalho apresentou três novos *workflows* científicos de diferentes áreas, sendo um da área de Astronomia, um da área de Bioquímica e, por fim, um da área de Reconhecimento Facial (ou Computação Visual), e executou-os na plataforma BioNimbuZ, que garantiu a realização de suas tarefas e obtenção de seus resultados finais, garantindo assim que a plataforma permite executar *workflows* científico de qualquer área.

Assim, o BioNimbuZ muda de patamar, deixando de ser considerado uma plataforma para execução de *workflow* de Bioinformática, para tornar-se uma plataforma para execução de qualquer *workflow* científico, realizando pouca ou nenhuma alteração. Apesar da necessidade de implantar o *script* das tarefas na plataforma, as maiores adaptações realizadas estão relacionadas ao fato da utilização de bibliotecas nas tarefas do *workflow* e ao tratamento de pastas, pois o sistema não permite o *upload/download* de diretórios.

A plataforma também permitiu usufruir dos benefícios do modelo de computação em nuvem federada, acessando as máquinas virtuais disponíveis no momento da configuração de cada tarefa durante a implantação no sistema. Assim, é possível realizar uma estimativa de processamento e de armazenamento necessário para a realização da tarefa, além de apresentar o custo pelo serviço, cobrado pelo provedor.

Dentre as possibilidades de trabalhos futuros ou continuidade deste trabalho, é proposto a busca por novos *workflows* científicos de outras áreas além das propostas neste trabalho, exigindo da plataforma novos casos de execução. É necessário também verificar os gargalos existentes na plataforma, e formular uma maneira em que a criação de maquinas virtuais demore menos. Também surge como trabalho futuro a integração de um

sistema de contêineres à plataforma, adicionar a possibilidade de modificação de *workflows*, e também uma opção para salvar o diagrama de um *workflow* como arquivo, para ser reutilizado. Além disso, pretende-se executar *workflows* dinâmicos, os quais podem ter novas tarefas sendo criadas durante a execução do *workflow*.

Referências

[1] Juve, Gideon, Ann Chervenak, Ewa Deelman, Shishir Bharathi, Gaurang Mehta e Karan Vahi: *Characterizing and profiling scientific workflows*. Future Generation Computer Systems, 29(3):682–692, 2013. ix, 5, 6

[2] Bharathi, Shishir, Ann Chervenak, Ewa Deelman, Gaurang Mehta, Mei Hui Su e Karan Vahi: *Characterization of scientific workflows*. Em *Workflows in Support of Large-Scale Science, 2008. WORKS 2008. Third Workshop on*, páginas 1–10. IEEE, 2008. ix, 6, 7

[3] Ramos, Vinícius de Almeida: *Um sistema gerenciador de workflows científicos para a plataforma de nuvens federadas bionimbuz*. Monografia de graduação, Departamento de Ciência de Computação, Universidade de Brasília, 2016. `http://repositorio.unb.br`. ix, 4, 8, 27, 28

[4] Rosa, Michel Junio Ferreira: *Predição de tempo e dimensionamento de recursos para workflows científicos em nuvens federadas*. Tese de Mestrado, Programa de Pós-Graduação em Informática, Departamento de Ciência de Computação, Universidade de Brasília, 2017. ix, 2, 8, 9, 23, 24, 27

[5] Paula, Renato de: *Proveniência de dados em workflows de bioinformática*. Tese de Mestrado, Programa de Pós-Graduação em Informática, Departamento de Ciência de Computação, Universidade de Brasília, 2013. ix, 9, 10

[6] Berriman, G Bruce e John C Good: *Sustaining the montage image mosaic engine since 2002*. arXiv preprint arXiv:1806.04095, 2018. ix, 10, 11

[7] Technology, California Institute of: *Montage user documentation*. `http://montage.ipac.caltech.edu/docs/mViewer.html`. Acessado em: 15-11-2018. ix, 12

[8] Delker, Silvia L, Anthony P West Jr, Lindsay McDermott, Malcolm W Kennedy e Pamela J Bjorkman: *1t80*. `https://www.rcsb.org/structure/1T80/`. Acessado em: 10-12-2018. ix, 13, 14, 41

[9] Minichino, Joe e Joseph Howse: *Learning OpenCV 3 Computer Vision with Python*. Packt Publishing Ltd, 2015. ix, 15, 16, 38, 47

[10] Souza, Felipe Lopes de: *Bionimbuz 2 - uma plataforma de federação de nuvens em uma arquitetura orientada a microsserviços*. Tese de Mestrado, Programa de Pós-Graduação em Informática, Departamento de Ciência de Computação, Universidade de Brasília, 2018. ix, 2, 19, 22, 25, 27, 28, 29, 31, 32, 33, 37

[11] Toosi, Adel Nadjaran, Rodrigo N Calheiros e Rajkumar Buyya: *Interconnected cloud computing environments: Challenges, taxonomy, and survey.* ACM Computing Surveys (CSUR), 47(1):7, 2014. ix, 20, 21, 22, 25

[12] Celesti, Antonio, Francesco Tusa, Massimo Villari e Antonio Puliafito: *How to enhance cloud architectures to enable cross-federation.* Em *2010 IEEE 3rd international conference on cloud computing*, páginas 337–345. IEEE, 2010. ix, 1, 2, 20, 21, 22, 23, 24, 25, 26

[13] Buyya, Rajkumar, Rajiv Ranjan e Rodrigo N Calheiros: *Intercloud: Utility-oriented federation of cloud computing environments for scaling of application services.* Em *International Conference on Algorithms and Architectures for Parallel Processing*, páginas 13–31. Springer, 2010. ix, 2, 22, 23, 24, 26

[14] Gomes, Jefferson Chaves: *Modelo multi-estratégico de tolerância a falhas para ambiente de nuvem federada.* Tese de Mestrado, Programa de Pós-Graduação em Informática, Departamento de Ciência de Computação, Universidade de Brasília, 2018. ix, 1, 2, 17, 22, 25, 27, 28, 29, 32

[15] Mell, Peter, Tim Grance *et al.*: *The nist definition of cloud computing.* 2011. 1, 17, 18, 19

[16] Knorr, Eric e Galen Gruman: *What cloud computing really means.* InfoWorld, 7:20–20, 2008. 1, 17

[17] Saldanha, Hugo Vasconcelos: *Bionimbus: uma arquitetura de federação de nuvens computacionais híbrida para a execução de workflows de bioinformática.* Tese de Mestrado, Programa de Pós-Graduação em Informática, Departamento de Ciência de Computação, Universidade de Brasília, 2012. 2, 22, 26, 27, 37

[18] Gil, Yolanda, Ewa Deelman, Mark Ellisman, Thomas Fahringer, Geoffrey Fox, Dennis Gannon, Carole Goble, Miron Livny, Luc Moreau e Jim Myers: *Examining the challenges of scientific workflows.* Computer, 40(12), 2007. 2, 4

[19] Moura, Breno Rodrigues e Deric Lima Bacelar: *Política para armazenamento de arquivos no zoonimbus.* Monografia de graduação, Departamento de Ciência de Computação, Universidade de Brasília, 2013. `http://repositorio.unb.br`. 2, 27

[20] Moura, Breno Rodrigues de: *Arquitetura de um controlador de sla para ambiente de nuvens federadas.* Tese de Mestrado, Programa de Pós-Graduação em Informática, Departamento de Ciência de Computação, Universidade de Brasília, 2017. `http://repositorio.unb.br`. 2, 27, 28

[21] Vergara, Guilherme Fay: *Arquitetura de um controlador de elasticidade para nuvens federadas.* 2, 27, 28

[22] Braghetto, Kelly Rosa e Daniel Cordeiro: *Introdução à modelagem e execução de workflows científicos.* Atualizações em Informática. 1ed. Porto Alegre: SBC, páginas 1–40, 2014. 4, 7

[23] Singh, Munindar P e Mladen A Vouk: *Scientific workflows: scientific computing meets transactional workflows*. Em *Proceedings of the NSF Workshop on Workflow and Process Automation in Information Systems: State-of-the-Art and Future Directions*, páginas 28–34, 1996. 4

[24] Watanabe, E, Pedro PV Campos, K Braghetto e D Batista: *Algoritmos para economia de energia no escalonamento de workflows em nuvens computacionais*. Em *32nd Brazilian Symposium on Computer Networks and Distributed Systems, Florianopolis, Brazil*, 2014. 4, 10

[25] Maechling, Philip, Ewa Deelman, Li Zhao, Robert Graves, Gaurang Mehta, Nitin Gupta, John Mehringer, Carl Kesselman, Scott Callaghan, David Okaya *et al.*: *Scec cybershake workflows—automating probabilistic seismic hazard analysis calculations*. Em *Workflows for e-Science*, páginas 143–163. Springer, 2007. 5

[26] McGuire, Robin K: *Probabilistic seismic hazard analysis and design earthquakes: closing the loop*. Bulletin of the Seismological Society of America, 85(5):1275–1284, 1995. 5

[27] Southern California, University of: *Usc molecular genomics core*. Acessado em: 10-12-2018. 5

[28] Lab, Hannon: *Fastx-toolkit*. `http://hannonlab.cshl.edu/fastx_toolkit/`. Acessado em: 15-11-2018. 9

[29] Langmead, Ben, Cole Trapnell, Mihai Pop e Steven L Salzberg: *Ultrafast and memory-efficient alignment of short dna sequences to the human genome*. Genome biology, 10(3):R25, 2009. 9

[30] Technology, California Institute of: *Montage - image mosaic software for astronomers*. `http://montage.ipac.caltech.edu/`. Acessado em: 15-11-2018. 10

[31] LA, REFORMAS EN e ADMINISTRACIÓN DE JUSTICIA: *Cr ó nica*. Boletin informativo, 2009. 10

[32] Berriman, G Bruce e JC Good: *The application of the montage image mosaic engine to the visualization of astronomical images*. Publications of the Astronomical Society of the Pacific, 129(975):058006, 2017. 10

[33] Corporation, Microsoft: *Windows | site oficial do sistema operacional microsoft windows 10 home e pro, laptops, computadores, tablets e muito mais*. `https://www.microsoft.com/pt-br/windows`. Acessado em: 15-11-2018. 10

[34] Foundation, Python Software: *Welcome to python.org*. `https://www.python.org/`. Acessado em: 15-11-2018. 10, 13

[35] Technology, California Institute of: *Montage user documentation*. `http://montage.ipac.caltech.edu/docs/first_mosaic_tutorial.html`. Acessado em: 15-11-2018. 11

[36] Parker, Steve: *The shell scripting tutorial*. `https://www.shellscript.sh/`. Acessado em: 10-11-2018. 11

[37] Stark, J Alex: *Adaptive image contrast enhancement using generalizations of histogram equalization*. IEEE Transactions on image processing, 9(5):889–896, 2000. 12

[38] Guedes, Fabiana Costa, Carlos Henrique Silveira e Aleteia Patrícia Favacho de Araújo: *Utilização do bionimbuz para desenvolvimento de workflows em dinâmica molecular*. página 4, 2018. 13

[39] Viana, Dianne, Julia Zamboni, André Paiva, Gabriel Gaspar, Flavio Vidal, João Gomes, Felipe Ferreira e Nathan Souza: *Integrando arte, computação e engenharia no desenvolvimento e construção de um personagem robótico*. XL Congresso Brasileiro de Educação em Engenharia, página 12, 2012. 14, 15, 16

[40] Penharbel, Eder Augusto, Erdiane LG Wutzke, Murilo dos S Silva e Reinaldo AC Bianchi: *E-faces-um classificador capaz de analisar imagens e classifica-las como faces ou nao faces utilizando o método eigenfaces*. IW orkshop de Visão Computacional, página 13, 2005. 16

[41] Ruschel, Henrique, Mariana Susan Zanotto e WC da Mota: *Computação em nuvem*. Curitiba, abr, páginas 1–3, 2010. 17

[42] Foster, Ian, Yong Zhao, Ioan Raicu e Shiyong Lu: *Cloud computing and grid computing 360-degree compared*. Em *2012 ACM/IEEE 13TH INTERNATIONAL CONFERENCE ON GRID COMPUTING*. Citeseer, 2012. 18

[43] Pedrosa, Paulo HC e Tiago Nogueira: *Computação em nuvem*. artigo disponível em http://www. ic. unicamp. br/˜ ducatte/mo401/1s2011, 2, 2011. 19, 20

[44] Wang, Jianwu, Moustafa AbdelBaky, Javier Diaz-Montes, Shweta Purawat, Manish Parashar e Ilkay Altintas: *Kepler+ cometcloud: dynamic scientific workflow execution on federated cloud resources*. Procedia Computer Science, 80:700–711, 2016. 20

[45] Barril, Jose Farnesio Huesca, Jeff Ruyter e Qing Tan: *A view on internet of things driving cloud federation*. Em *Cloud Computing and Big Data Analysis (ICCCBDA), 2016 IEEE International Conference on*, páginas 221–226. IEEE, 2016. 20

[46] Baghban, Hojjat, Mahdis Moradi, Ching Hsien Hsu, Jerry Chou e Yeh Ching Chung: *Byzantine fault tolerant optimization in federated cloud computing*. Em *Computer and Information Technology (CIT), 2016 IEEE International Conference on*, páginas 658–661. IEEE, 2016. 20

[47] Goiri, Inigo, Jordi Guitart e Jordi Torres: *Characterizing cloud federation for enhancing providers' profit*. Em *Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on*, páginas 123–130. IEEE, 2010. 21

[48] Foundation, The Apache Software: *Apache zookeeper - home*. `https://zookeeper.apache.org/`. Acessado em: 22-11-2018. 27, 32

[49] Foundation, The Apache Software: *Welcome to apache avro*. `https://avro.apache.org/`. Acessado em: 22-11-2018. 27

[50] Oliveira, Gabriel SS de, Edward Ribeiro, Diogo A Ferreira, Aletéia PF Araújo, Maristela T Holanda e Maria Emilia MT Walter: *Acosched: A scheduling algorithm in a federated cloud infrastructure for bioinformatics applications*. Em *Bioinformatics and Biomedicine (BIBM), 2013 IEEE International Conference on*, páginas 8–14. IEEE, 2013. 27

[51] Azevedo, DR e TB Freitas Júnior: *Uma nova política de armazenamento para a plataforma bionimbuz de nuvem federada*. Monografia de graduação, Departamento de Ciência de Computação, Universidade de Brasília, 2015. `http://repositorio.unb.br`. 27

[52] Deelman, Ewa, Gurmeet Singh, Mei Hui Su, James Blythe, Yolanda Gil, Carl Kesselman, Gaurang Mehta, Karan Vahi, G Bruce Berriman, John Good *et al.*: *Pegasus: A framework for mapping complex scientific workflows onto distributed systems*. Scientific Programming, 13(3):219–237, 2005. 28

[53] Foundation, The Apache Software: *Apache airflow documentation - airflow documentation*. `http://airflow.apache.org/index.html`. 28

[54] Dmitry, Namiot e Sneps Sneppe Manfred: *On micro-services architecture*. International Journal of Open Information Technologies, 2(9), 2014. 28, 32

[55] Technology, California Institute of: *Irsa*. `http://irsa.ipac.caltech.edu/`. Acessado em: 10-12-2018. 41